

Chapter 7

Configuring Protocol Family and Address Interface Properties

For each logical interface, you must configure one or more protocol families. You can also configure interface address properties. To do this, you include the following statements:

```
family family {
    accounting {
        destination-class-usage;
        source-class-usage {
            (input | output | input output);
        }
    }
    bundle interface-name;
    filter {
        input filter-name;
        output filter-name;
        group filter-group-number;
    }
    ipsec-sa sa-name;
    keep-address-and-control;
    mtu bytes;
    multicast-only;
    negotiate-address;
    no-redirects;
    policer {
        arp policer-template-name;
        input policer-template-name;
        output policer-template-name;
    }
    primary;
    proxy inet-address address;
    receive-options-packets;
    receive-ttl-exceeded;
    remote (inet-address address | mac-address address);
    rpf-check <fail-filter filter-name> {
        <mode loose>;
    }
    sampling {
        (input | output | input output);
    }
}
```

```

service {
  input {
    [ service-set service-set-name <service-filter filter-name> ];
    post-service-filter filter-name;
  }
  output {
    [ service-set service-set-name <service-filter filter-name> ];
  }
}
(translate-discard-eligible | no-translate-discard-eligible);
(translate-fecn-and-becn | no-translate-fecn-and-becn);
unnumbered-address interface-name destination address
  destination-profile profile-name;
address address {
  arp ip-address (mac | multicast-mac) mac-address <publish>;
  broadcast address;
  destination address;
  destination-profile name;
  eui-64;
  multipoint-destination address (dlsi dlsi-identifier |
    vci vci-identifier);
  multipoint-destination address {
    epd-threshold cells plp1 cells;
    inverse-arp;
    oam-liveness {
      up-count cells;
      down-count cells;
    }
    oam-period (disable | seconds);
    shaping {
      (cbr rate | rtvbr peak rate sustained rate burst length |
        vbr peak rate sustained rate burst length);
      queue-length number;
    }
    vci vpi-identifier.vci-identifier;
  }
}
preferred;
primary;
vrrp-group group-number {
  (accept-data | no-accept-data);
  advertise-interval seconds;
  authentication-type authentication;
  authentication-key key;
  fast-interval milliseconds;
  (preempt | no-preempt) {
    hold-time seconds;
  }
  priority number;
  track {
    interface interface-name priority-cost cost;
  }
  virtual-address [ addresses ];
}
}
}

```

You can include these statements at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit  
logical-unit-number]
```

This chapter describes the configuration of the interface protocol and address properties:

Configuring the Protocol Family on page 110

Configuring the Interface Address on page 112

Configuring IPCP Options on page 114

Configuring an Unnumbered Interface on page 117

Setting the Protocol MTU on page 117

Disabling the Removal of Address and Control Bytes on page 118

Disabling the Transmission of Redirect Messages on an Interface on page 119

Configuring Default, Primary, and Preferred Addresses and Interfaces on page 119

Applying Policers on page 121

Applying a Filter to an Interface on page 122

Configuring Unicast RPF on page 124

Enabling Source Class and Destination Class Usage on page 130

For information about interface-specific protocol and address properties, see “Interface Types” on page 155.

Configuring the Protocol Family

For each logical interface, you can configure one or more of the following protocols that run on the interface:

ccc—Circuit cross-connect (CCC). You can configure this protocol family for the logical interface of CCC physical interfaces. When you use this encapsulation type, you can configure the ccc family only.

inet—IP. You must configure this protocol family for the logical interface to support IP protocol traffic, including Open Shortest Path First (OSPF), Border Gateway Protocol (BGP), Internet Control Message Protocol (ICMP), and Internet Protocol Control Protocol (IPCP).

inet6—IP version 6 (IPv6). You must configure this protocol family for the logical interface to support IPv6 protocol traffic, including Routing Information Protocol for IPv6 (RIPng), Intermediate System-to-Intermediate System (IS-IS), and BGP. For more information about IPv6, see “IPv6 Introduction” on page 111.

iso—International Organization for Standardization (ISO). You must configure this protocol family for the logical interface to support IS-IS traffic.

mlfr-uni-nni—Multilink Frame Relay (MLFR) FRF.16 user-to-network network-to-network (UNI NNI). You must configure this protocol or **mlfr-end-to-end** for the logical interface to support link services and voice services bundling.

mlfr-end-to-end—Multilink Frame Relay end-to-end. You must configure this protocol or multilink Point-to-Point Protocol (MLPPP) for the logical interface to support multilink bundling.

mlppp—MLPPP. You must configure this protocol (or **mlfr-end-to-end**) for the logical interface to support multilink bundling.

mpls—Multiprotocol Label Switching (MPLS). You must configure this protocol family for the logical interface to participate in an MPLS path.

tcc—Translational cross-connect (TCC). You can configure this protocol family for the logical interface of TCC physical interfaces.

tnp—Trivial Network Protocol. This protocol is used to communicate between the Routing Engine and the routing platform’s packet forwarding components. The JUNOS software automatically configures this protocol family on the routing platform’s internal interfaces only, as discussed in “Displaying the Internal Ethernet Interface” on page 391.

vpls—Virtual private LAN service (VPLS). You can optionally configure this protocol family for the logical interface on which you configure VPLS. VPLS provides an Ethernet-based point-to-multipoint Layer 2 VPN to connect customer edge (CE) routing platforms across an MPLS backbone. When you configure a VPLS encapsulation type, the family **vpls** statement is assumed by default. For more information about VPLS, see the *JUNOS VPNs Configuration Guide* and the *JUNOS Feature Guide*.

To configure the logical interface's protocol family, include the family statement, specifying the selected family. To configure more than one protocol family on a logical interface, include multiple family statements. Following is the minimum configuration:

```
family family {
  mtu size;
  multicast-only;
  no-redirects;
  primary;
  address address {
    destination address;
    broadcast address;
    preferred;
    primary;
  }
}
```

You can include these statements at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number]
```

IPv6 Introduction

IP version 4 (IPv4) has been widely deployed and used to network the Internet today. With the rapid growth of the Internet, enhancements to IPv4 are needed to support the influx of new subscribers, Internet-enabled devices, and applications. IPv6 is designed to enable the global expansion of the Internet.

IPv6 builds upon the functionality of IPv4, providing improvements to addressing, configuration and maintenance, and security.

IPv6 is defined in the following documents:

RFC 2373, *IP Version 6 Addressing Architecture*

RFC 2460, *Internet Protocol, Version 6 (IPv6)*

IPv4-to-IPv6 Transition

Implementing IPv6 requires a transition mechanism to allow interoperability between IPv6 nodes (both routing platforms and hosts) and IPv4 nodes. The transition mechanism is the key factor in the successful deployment of IPv6. Because millions of IPv4 nodes already exist, upgrading every node to IPv6 at the same time is not feasible.

As a result, transition from IPv4 to IPv6 happens gradually, allowing nodes to be upgraded independently and without disruption to other nodes. While a gradual upgrade occurs, compatibility between IPv6 and IPv4 nodes becomes a requirement. Otherwise, an IPv6 node would not be able to communicate with an IPv4 node.

Transition mechanisms allow IPv6 and IPv4 nodes to coexist together in the same network, and make gradual upgrading possible. The transition mechanism supported by the JUNOS Internet software is tunneling. Tunnels allow IPv6 packets to be encapsulated into IPv4 headers and sent across an IPv4 infrastructure. For more information about configuring tunnels to support IPv4-to-IPv6 transition, see “Configuring an IPv6-over-IPv4 Tunnel” on page 578.

Configuring the Interface Address

You assign an address to an interface by specifying the address when configuring the protocol family. For the inet family, you configure the interface’s IP address. For the iso family, you configure one or more addresses for the loopback interface. For the ccc, tcc, mpls, tnp, and vpls families, you never configure an address.

To assign an address to an interface, include the address statement:

```
address address {
    broadcast address;
    destination address;
    destination-profile name;
    eui-64;
    preferred;
    primary;
}
```

You can include these statements at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family]
```

In the address statement, specify the network address of the interface.

For each address, you can optionally configure one or more of the following:

Broadcast address for the interface's subnet—Specify this in the broadcast statement; this applies only to Ethernet interfaces, such as the management interface `fxp0`, the Fast Ethernet interface, and the Gigabit Ethernet interface.

Address of the remote side of the connection (for point-to-point interfaces only)—Specify this in the destination statement.

Assign PPP properties to the remote end—Specify this in the destination-profile statement. You define the profile at the [edit access group-profile *name* ppp] hierarchy level (for point-to-point interfaces only). For more information, see “Configuring IPCP Options” on page 114.

Whether the routing platform automatically generates the host number portion of interface addresses—The `eui-64` statement applies only to interfaces that carry IPv6 traffic, where the prefix length of the address is 64 bits or less, and the low-order 64 bits of the address are zero. This option does not apply to the loopback interface (`lo0`) because IPv6 addresses configured on the loopback interface must have a 128-bit prefix length.

Whether this address is the preferred address—Each subnet on an interface has a preferred local address. If you configure more than one address on the same subnet, the preferred local address is chosen by default as the source address when you originate packets to destinations on the subnet. For more information about preferred addresses, see “Configuring Default, Primary, and Preferred Addresses and Interfaces” on page 119.

By default, the preferred address is the lowest numbered address on the subnet. To override the default and explicitly configure the preferred address, include the preferred statement when configuring the address.

Whether this address is the primary address—Each interface has a primary local address. If an interface has more than one address, the primary local address is used by default as the source address when you originate packets out the interface where the destination gives no hint about the subnet (for example, some ping commands). For more information about primary addresses, see “Configuring Default, Primary, and Preferred Addresses and Interfaces” on page 119.

By default, the primary address on an interface is the lowest numbered non-127 preferred address on the interface. To override the default and explicitly configure the preferred address, include the primary statement when configuring the address.

Configuring the IPv6 Address on an Interface

You represent IPv6 addresses in hexadecimal notation using a colon-separated list of 16-bit values.

You assign a 128-bit IPv6 address to an interface by including the address statement:

```
address aaaa:bbbb:...:zzzz/nn;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family inet6]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit logical-unit-number family inet6]
```

The double colon (::) represents all bits set to 0, as shown in the following example:

```
interfaces fe-0/0/1 {
  unit 0 {
    family inet6 {
      address fec0:1:1:1::2/64;
    }
  }
}
```

Configuring IPCP Options

For interfaces with PPP encapsulation, you can configure IPCP to negotiate IP address assignments and to pass network-related information such as Windows Name Service (WINS) and Domain Name System (DNS) servers, as defined in RFC 1877, *PPP Internet Protocol Control Protocol Extensions for Name Server Addresses*.



NOTE: The JUNOS software does not request name servers from the remote end; the software does, however, send name servers to the remote end if requested.

On the logical interface, the following PPP encapsulation types are supported:

```
atm-mlppp-llc
atm-ppp-llc
atm-ppp-vc-mux
multilink-ppp
```

When you enable a PPP interface, you can configure an IP address, enable the interface to negotiate an IP address assignment from the remote end, or allow the interface to be unnumbered. You can also assign a destination profile to the remote end. The destination profile includes PPP properties, such as primary and secondary DNS and NetBIOS Name Servers (NBNSs). These options are described in the following sections:

Configuring an IP Address for an Interface on page 115

Negotiating an IP Address Assignment from the Remote End on page 115

Configuring an Interface to Be Unnumbered on page 115

Assigning a Destination Profile to the Remote End on page 116

Configuring an IP Address for an Interface

You can configure an IP address for the interface by including the address statement in the configuration. For more information, see “Configuring the Interface Address” on page 112.

If you include the address statement in the configuration, you cannot include the `negotiate-address` or `unnumbered-address` statement in the configuration.

When you include the address statement in the interface configuration, you can assign PPP properties to the remote end, as shown in “Assigning a Destination Profile to the Remote End” on page 116.

Negotiating an IP Address Assignment from the Remote End

To enable the interface to obtain an IP address from the remote end, include the `negotiate-address` statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet] hierarchy level:

```
negotiate-address;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family inet]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family inet]
```

If you include the `negotiate-address` statement in the configuration, you cannot include the `address` or `unnumbered-address` statement in the configuration.

Configuring an Interface to Be Unnumbered

To configure an interface to be unnumbered, include the `unnumbered-address` and `destination` statements in the configuration:

```
unnumbered-address interface-name destination address;
```

The `unnumbered-address` statement enables the local address to be derived from the specified interface. The interface name must include a logical unit number and must have a configured address (see “Configuring the Interface Address” on page 112). Specify the IP address of the remote interface with the `destination` statement.

You can include these statements at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family inet]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family inet]
```

If you include the `unnumbered-address` statement in the configuration, you cannot include the `address` or `negotiate-address` statement in the interface configuration.

When you include the `unnumbered-address` statement in the interface configuration, you can assign PPP properties to the remote end, as shown in “Assigning a Destination Profile to the Remote End” on page 116.

Assigning a Destination Profile to the Remote End

When you include the `address` or `unnumbered-address` statement in the interface configuration, you can assign PPP properties to the remote end. To do this, include the `destination-profile` statement:

```
destination-profile name;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family inet address address]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family inet address address]
```

```
[edit interfaces interface-name unit logical-unit-number family inet
unnumbered-address interface-name]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family inet unnumbered-address interface-name]
```

The profile name is a PPP group profile. You define the profile by including the following statements at the `[edit access group-profile name ppp]` hierarchy level:

```
[edit access group-profile name ppp]
framed-pool pool-id;
interface-id interface-id;
primary-dns primary-dns;
primary-wins primary-win-server;
secondary-dns secondary-dns;
secondary-wins secondary-wins;
```

For more information about PPP group profiles, see the *JUNOS System Basics Configuration Guide*.

Configuring an Unnumbered Interface

When you need to conserve IP addresses, you can configure unnumbered interfaces. To do this, configure the protocol family, but do not include the address statement:

```
family family;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number]
```



NOTE: For interfaces with PPP encapsulation, you can configure an unnumbered interface by including the unnumbered-interface statement in the configuration. For more information, see “Configuring IPCP Options” on page 114.

When configuring unnumbered interfaces, you must ensure that a source address is configured on some interface in the routing platform. This address is the default address. We recommend that you do this by assigning an address to the loopback interface (lo0), as described in “Configuring the Loopback Interface” on page 485. If you configure an address (other than a martian) on the lo0 interface, that address is always the default address, which is preferable because the loopback interface is independent of any physical interfaces and therefore is always accessible.

Example: Configuring an Unnumbered Interface

Configure an unnumbered interface:

```
[edit]
interfaces {
  so-6/1/0 {
    unit 0 {
      family inet;
      family iso;
    }
  }
}
```

Setting the Protocol MTU

When you initially configure an interface, the protocol maximum transmission unit (MTU) is calculated automatically. If you subsequently change the media MTU, the protocol MTU on existing address families automatically changes.

For a list of default protocol MTU values, see “Configuring the Media MTU” on page 67.

To modify the MTU for a particular protocol family, include the mtu statement:

```
mtu bytes;
```

You can include this statement at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family]
```

If you increase the size of the protocol MTU, you must ensure that the size of the media MTU is equal to or greater than the sum of the protocol MTU and the encapsulation overhead. For a list of encapsulation overhead values, see Table 11 on page 71. If you reduce the media MTU size, but there are already one or more address families configured and active on the interface, you must also reduce the protocol MTU size. (You configure the media MTU by including the `mtu` statement at the `[edit interfaces interface-name]` hierarchy level, as discussed in “Configuring the Media MTU” on page 67.)

The maximum number of data-link connection identifiers (DLCIs) is determined by the MTU on the interface. If you have keepalives enabled, the maximum number of DLCIs is 1000, with the MTU set to 5012.

The actual frames transmitted also contain cyclic redundancy check (CRC) bits, which are not part of the MTU. For example, the default protocol MTU for a Gigabit Ethernet interface is 1500 bytes, but the largest possible frame size is actually 1504 bytes; you need to consider the extra bits in calculations of MTUs for interoperability.

Disabling the Removal of Address and Control Bytes

For Point-to-Point Protocol (PPP) CCC-encapsulated interfaces, the address and control bytes are removed by default before the packet is encapsulated into a tunnel.

You can disable the removal of address and control bytes. To do this, include the `keep-address-and-control` statement:

```
keep-address-and-control;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family ccc]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family ccc]
```

Disabling the Transmission of Redirect Messages on an Interface

By default, the interface sends protocol redirect messages. To disable the sending of these messages on an interface, include the `no-redirects` statement:

```
no-redirects;
```

You can include this statement at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family]
```

To disable the sending of protocol redirect messages for the entire routing platform, include the `no-redirects` statement at the `[edit system]` hierarchy level.

Configuring Default, Primary, and Preferred Addresses and Interfaces

The routing platform has a default address and a primary interface, and interfaces have primary and preferred addresses.

The *default address* of the routing platform is used as the source address on unnumbered interfaces. The routing protocol process tries to pick the default address as the routing platform ID, which is used by protocols, including OSPF and internal BGP (IBGP).

The *primary interface* for the routing platform is the interface that packets go out when no interface name is specified and when the destination address does not imply a particular outgoing interface.

An interface's *primary address* is used by default as the local address for broadcast and multicast packets sourced locally and sent out the interface. An interface's *preferred address* is the default local address used for packets sourced by the local routing platform to destinations on the subnet.

The default address of the routing platform is chosen using the following sequence:

1. The primary address on the loopback interface `lo0` that is not `127.0.0.1` is used.
2. The primary address on the primary interface is used.

To configure these addresses and interfaces, you can do the following:

Configuring the Primary Interface for the Routing Platform on page 120

Configuring the Primary Address for an Interface on page 120

Configuring the Preferred Address for an Interface on page 121

Configuring the Primary Interface for the Routing Platform

The *primary interface* for the routing platform has the following characteristics:

It is the interface that packets go out when you type a command such as ping 255.255.255.255—that is, a command that does not include an interface name (there is no interface *type-0/0/0.0* qualifier) and where the destination address does not imply any particular outgoing interface.

It is the interface on which multicast applications running locally on the routing platform, such as Session Announcement Protocol (SAP), do group joins by default.

It is the interface from which the default local address is derived for packets sourced out an unnumbered interface if there are no non-127 addresses configured on the loopback interface, lo0.

By default, the multicast-capable interface with the lowest-index address is chosen as the primary interface. If there is no such interface, the point-to-point interface with the lowest index address is chosen. Otherwise, any interface with an address could be picked. In practice, this means that, on the routing platform, the fxp0 interface is picked by default.

To configure a different interface to be the primary interface, include the primary statement:

```
primary;
```

You can include this statement at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit logical-unit-number family family]
```

Configuring the Primary Address for an Interface

The *primary address* on an interface is the address that is used by default as the local address for broadcast and multicast packets sourced locally and sent out the interface. For example, the local address in the packets sent by a ping interface so-0/0/0.0 255.255.255.255 command is the primary address on interface so-0/0/0.0. The primary address flag also can be useful for selecting the local address used for packets sent out unnumbered interfaces when multiple non-127 addresses are configured on the loopback interface, lo0. By default, the primary address on an interface is selected as the numerically lowest local address configured on the interface.

To set a different primary address, include the primary statement:

```
primary;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family family address
address]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family address address]
```

Configuring the Preferred Address for an Interface

The *preferred address* on an interface is the default local address used for packets sourced by the local routing platform to destinations on the subnet. By default, the numerically lowest local address is chosen. For example, if the addresses 172.16.1.1/12, 172.16.1.2/12, and 172.16.1.3/12 are configured on the same interface, the preferred address on the subnet (by default, 172.16.1.1) would be used as a local address when you issue a ping 172.16.1.5 command.

To set a different preferred address for the subnet, include the preferred statement:

```
preferred;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family family address
address]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family address address]
```

Applying Policers

Policers allow you to perform simple traffic policing on specific interfaces or Layer 2 virtual private networks (VPNs) without configuring a firewall filter. To apply policers, include the policer statement:

```
policer {
    arp policer-template-name;
    input policer-template-name;
    output policer-template-name;
}
```

You can include these statements at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family]
```

In the family statement, the protocol family can be ccc, inet, tcc, or vpls.

In the `arp` statement, list the name of one policer template to be evaluated when Address Resolution Protocol (ARP) packets are received on the interface. By default, an ARP policer is installed that is shared among all the Ethernet interfaces on which you have configured the `family inet` statement. If you want more stringent or lenient policing of ARP packets, you can configure an interface-specific policer and apply it to the interface. You configure an ARP policer just as you would configure any other policer, at the `[edit firewall policer]` hierarchy level. If you apply this policer to an interface, the default ARP packet policer is overridden. If you delete this policer, the default policer takes effect again.

In the `input` statement, list the name of one policer template to be evaluated when packets are received on the interface.

In the `output` statement, list the name of one policer template to be evaluated when packets are transmitted on the interface.



NOTE: To use policing on a CCC or TCC interface, you must configure the CCC or TCC protocol family.

You can configure a different policer on each protocol family on an interface, with one input policer and one output policer for each family. When you apply policers, you can configure the `family cc`, `inet`, `tcc`, or `vpls` only, and one ARP policer for the `family inet` protocol only. Each time a policer is referenced, a separate copy of the policer is installed on the packet forwarding components for that interface.

If you apply both policers and firewall filters to an interface, input policers are evaluated before input firewall filters, and output policers are evaluated after output firewall filters.

If you apply the policer to the interface `lo0`, it is applied to packets received or transmitted by the Routing Engine.

For more information about policers, see the *JUNOS Policy Framework Configuration Guide*.

Applying a Filter to an Interface

To apply firewall filters to an interface, include the filter statement:

```
filter {
  group filter-group-number;
  input filter-name;
  output filter-name;
}
```

You can include these statements at the following hierarchy levels:

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

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family family]
```

In the family statement, the protocol family can be `inet`, `inet6`, `mpls`, or `vpls`.

In the group statement, specify the interface group number to associate with the filter.

In the input statement, list the name of one firewall filter to be evaluated when packets are received on the interface.

In the output statement, list the name of one firewall filter to be evaluated when packets are transmitted on the interface.

You can use the same filter one or more times.

For filter-based forwarding (FBF), you can configure input packet filters only; FBF is not supported for output filters.

If you apply the filter to the interface `lo0`, it is applied to packets received or transmitted by the Routing Engine. You cannot apply MPLS filters to the management interface (`fxp0`) or the loopback interface (`lo0`).

For more information about firewall filters, see the *JUNOS Policy Framework Configuration Guide*. For more information about MPLS filters, see the *JUNOS MPLS Applications Configuration Guide*. For more information about FBF, see the *JUNOS Routing Protocols Configuration Guide*.

Defining Interface Groups in Firewall Filters

When applying a firewall filter, you can define an interface to be part of an *interface group*. Packets received on that interface are tagged as being part of the group. You can then match these packets using the interface-group match statement, as described in the *JUNOS Policy Framework Configuration Guide*.

To define the interface to be part of an interface group, include the group statement:

```
group filter-group-number;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family family filter]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit logical-unit-number family family filter]
```

Configuring Unicast RPF

For interfaces that carry IPv4 or IPv6 traffic, you can reduce the impact of denial of service (DoS) attacks by configuring unicast reverse path forwarding (RPF). Unicast RPF helps determine the source of attacks and rejects packets from unexpected source addresses on interfaces where unicast RPF is enabled.



NOTE: If you want to configure unicast RPF, your routing platform must be equipped with the Internet Processor II application-specific integrated circuit (ASIC).

If you enable unicast RPF on live traffic, some packets are dropped while the packet forwarding components are updating.

For transit packets exiting the tunnel, forwarding path features, such as 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.

The following sections describe unicast RPF in detail:

Configuring Unicast RPF Strict Mode on page 124

Configuring Unicast RPF Loose Mode on page 125

Unicast RPF and Default Routes on page 126

Unicast RPF with Routing Asymmetry on page 127

Configuring Unicast RPF on a VPN on page 128

Example: Configuring Unicast RPF on page 129

Configuring Unicast RPF Strict Mode

In strict mode, unicast RPF checks whether the incoming packet has a source address that matches a prefix in the routing table, and whether the interface expects to receive a packet with this source address prefix.

If the incoming packet fails the unicast RPF check, the packet is not accepted on the interface. When a packet is not accepted on an interface, unicast RPF counts the packet and sends it to an optional fail filter.

The optional fail filter allows you to apply a filter to packets that fail the unicast RPF check. You can define the fail filter to perform any filter operation, including accepting, rejecting, logging, sampling, or policing.

When unicast RPF is enabled on an interface, Bootstrap Protocol (BOOTP) packets and Dynamic Host Configuration Protocol (DHCP) packets are not accepted on the interface. To allow the interface to accept BOOTP packets and DHCP packets, you must apply a fail filter that accepts all packets with a source address of 0.0.0.0 and a destination address of 255.255.255.255. For a configuration example, see “Example: Configuring Unicast RPF” on page 129.

For more information about defining fail filters, see the *JUNOS Policy Framework Configuration Guide*.

To configure unicast RPF, include the `rpf-check` statement:

```
rpf-check <fail-filter filter-name>;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family (inet | inet6)]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family (inet | inet6)]
```

Using unicast RPF can have several consequences when implemented with traffic filters:

RPF fail filters are evaluated after input filters and before output filters.

If you configure a filter counter for packets dropped by an input filter, and you want to know the total number of packets dropped, you must also configure a filter counter for packets dropped by the RPF check.

To count packets that fail the RPF check and are accepted by the RPF fail filter, you must configure a filter counter.

If an input filter forwards packets anywhere other than the `inet.0` or `inet6.0` routing tables, the unicast RPF check is not performed.

If an input filter forwards packets anywhere other than the routing instance the input interface is configured for, the unicast RPF check is not performed.

Configuring Unicast RPF Loose Mode

By default, unicast RPF uses strict mode. Unicast RPF loose mode is similar to unicast RPF strict mode and has the same configuration restrictions. The only check in loose mode is whether the packet has a source address with a corresponding prefix in the routing table; loose mode does not check whether the interface expects to receive a packet with a specific source address prefix. If a corresponding prefix is not found, unicast RPF loose mode does not accept the packet. As in strict mode, loose mode counts the failed packet and optionally forwards it to a fail filter, which either accepts, rejects, logs, samples, or polices the packet.

To configure unicast RPF loose mode, include the `mode`:

```
mode loose;
```

You can include this statement at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family (inet | inet6)
rpf-check <fail-filter filter-name>]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family (inet | inet6) rpf-check <fail-filter filter-name>]
```

Unicast RPF and Default Routes

When the active route cannot be chosen from the routes in a routing table, the routing platform chooses a default route. A default route is equivalent to an IP address of 0.0.0.0/0. If you configure a default route, and you configure unicast RPF on an interface that the default route uses, unicast RPF behaves differently than it does otherwise. For information about configuring default routes, see the *JUNOS Routing Protocols Configuration Guide*. The following sections describe how unicast RPF behaves when a default route uses an interface and when a default route does not use an interface:

Unicast RPF Behavior with a Default Route on page 126

Unicast RPF Behavior without a Default Route on page 127

To determine whether the default route uses an interface, enter the `show route` command:

```
user@host> show route address
```

`address` is the next-hop address of the configured default route. The default route uses the interfaces shown in the output of the `show route` command.

Unicast RPF Behavior with a Default Route

If you configure a default route that uses an interface configured with unicast RPF, unicast RPF behaves as follows:

Loose mode—All packets are automatically accepted. For this reason, we recommend that you not configure unicast RPF loose mode on interfaces that the default route uses.

Strict mode—The packet is accepted when either of the following is true:

- The source address of the packet matches any of the routes (either default or learned) that can be originated from the interface. Note that routes can have multiple destinations associated with them; therefore, if one of the destinations matches the incoming interface of the packet, the packet is accepted.

- The source address of the packet does not match any of the routes.

The packet is not accepted when either of the following is true:

- The source address of the packet does not match a prefix in the routing table.

- The interface does not expect to receive a packet with this source address prefix.

Unicast RPF Behavior without a Default Route

If you do not configure a default route, or if the default route does not use an interface configured with unicast RPF, unicast RPF behaves as described in “Configuring Unicast RPF Strict Mode” on page 124 and “Configuring Unicast RPF Loose Mode” on page 125. To summarize, unicast RPF without a default route behaves as follows:

Strict mode—The packet is not accepted when either of the following is true:

The packet has a source address that does not match a prefix in the routing table.

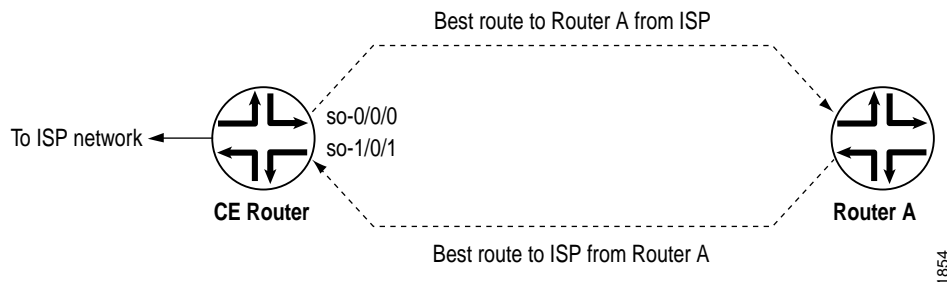
The interface does not expect to receive a packet with this source address prefix.

Loose mode—The packet is not accepted when the packet has a source address that does not match a prefix in the routing table.

Unicast RPF with Routing Asymmetry

In general, we recommend that you not enable unicast RPF on interfaces that are internal to the network because internal interfaces are likely to have *routing asymmetry*. Routing asymmetry means that a packet’s outgoing and return paths are different. Routers in the core of the network are more likely to have asymmetric reverse paths than routing platforms at the customer or provider edge. Figure 4 shows unicast RPF in an environment with routing asymmetry.

Figure 4: Unicast RPF with Routing Asymmetry



In Figure 4, if you enable unicast RPF on interface so-0/0/0, traffic destined for Router A is not rejected. If you enable unicast RPF on interface so-1/0/1, traffic from Router A is rejected.

If you need to enable unicast RPF in an asymmetric routing environment, you can use fail filters to allow the routing platform to accept incoming packets that are known to be arriving by specific paths. For an example of a fail filter that accepts packets with a specific source and destination address, see “Example: Configuring Unicast RPF” on page 129.

Configuring Unicast RPF on a VPN

You can configure unicast RPF on a VPN interface by enabling unicast RPF on the interface and including the interface statement at the [edit routing-instances *routing-instance-name*] hierarchy level.

You can configure unicast RPF only on the interfaces you specify in the routing instance. This means the following:

For Layer 3 VPNs, unicast RPF is supported on the CE router interface.

Unicast RPF is not supported on core-facing interfaces.

For virtual-router routing instances, unicast RPF is supported on all interfaces you specify in the routing instance.

If an input filter forwards packets anywhere other than the routing instance the input interface is configured for, the unicast RPF check is not performed.

For unicast RPF configuration guidelines, see “Configuring Unicast RPF” on page 124. For more information about VPNs and virtual-router routing instances, see the *JUNOS VPNs Configuration Guide*. For more information about FBF, see the *JUNOS Routing Protocols Configuration Guide*.

Example: Configuring Unicast RPF on a VPN

Configure unicast RPF on a Layer 3 VPN interface:

```
[edit interfaces]
so-0/0/0 {
  unit 0 {
    family inet {
      rpf-check;
    }
  }
}

[edit routing-instance]
VPN-A {
  interface so-0/0/0.0;
}
```

Example: Configuring Unicast RPF

Configure unicast RPF strict mode, and apply a fail filter that allows the interface to accept BOOTP packets and DHCP packets. The filter accepts all packets with a source address of 0.0.0.0 and a destination address of 255.255.255.255.

```
[edit firewall]
filter rpf-special-case-dhcp-bootp {
  term allow-dhcp-bootp {
    from {
      source-address {
        0.0.0.0/32;
      }
      address {
        255.255.255.255/32;
      }
    }
    then {
      count rpf-dhcp-bootp-traffic;
      accept;
    }
  }
  term default {
    then {
      log;
      reject;
    }
  }
}

[edit]
interfaces {
  so-0/0/0 {
    unit 0 {
      family inet {
        rpf-check fail-filter rpf-special-case-dhcp-bootp;
      }
    }
  }
}
```

Enabling Source Class and Destination Class Usage

For interfaces that carry IPv4, IPv6, or MPLS traffic, you can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into disjoint sets defined as *source classes* and *destination classes*. You can define classes based on a variety of parameters, such as routing neighbors, autonomous systems, and route filters.

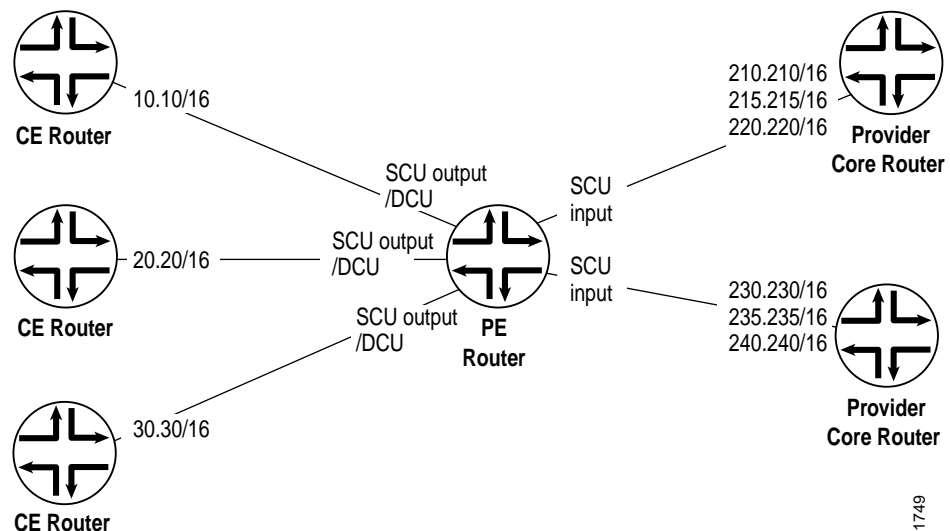
Source class usage (SCU) counts packets sent to customers by performing lookup on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge. You must enable SCU accounting on both the inbound and outbound physical interfaces.

Destination class usage (DCU) counts packets from customers by performing lookup of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

Figure 5 on page 130 illustrates an Internet service provider (ISP) network. In this topology, you can use DCU to count packets customers send to specific prefixes. For example, you can have three counters, one per customer, that count the packets destined for prefix 210.210/16 and 220.220/16.

You can use SCU to count packets the provider sends from specific prefixes. For example, you can count the packets sent from prefix 210.210/16 and 215.215/16 and transmitted on a specific output interface.

Figure 5: Prefix Accounting with Source and Destination Classes



1749

You can configure up to 126 source classes and 126 destination classes. For each interface on which you enable destination class usage and source class usage, the JUNOS software maintains an interface-specific counter for each corresponding class up to the 126 class limit.



NOTE: To configure source class and destination class usage, your routing platform must be equipped with the Internet Processor II ASIC.

For transit packets exiting the tunnel, forwarding path features, such as 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.

To enable packet counting on an interface, include the accounting statement:

```
accounting {
  destination-class-usage;
  source-class-usage {
    (input | output | input output);
  }
}
```

You can include these statements at the following hierarchy levels:

```
[edit interfaces interface-name unit logical-unit-number family (inet | inet6 | mpls)]
```

```
[edit logical-routers logical-router-name interfaces interface-name unit
logical-unit-number family (inet | inet6 | mpls)]
```

For SCU to work, you must configure at least one input interface and at least one output interface. An incoming packet is counted only once, and SCU takes priority over DCU. This means that when a packet arrives on an interface on which you include the source-class-usage input and destination-class-usage statements in the configuration, and when the source and destination both match accounting prefixes, the JUNOS software associates the packet with the source class only. To ensure the outgoing packet is counted, include the source-class-usage output statements in the configuration of the outgoing interface.

On T-series and M320 routing platforms, the source class and destination classes are not carried across the platform fabric. The implications of this are as follows:

On T-series and M320 platforms, SCU and DCU accounting is performed before the packet enters the fabric.

On T-series and M320 routing platforms, DCU is performed before output filters are evaluated. On other M-series platforms, DCU is performed after output filters are evaluated.

If an output filter drops traffic on T-series and M320 routing platforms, the dropped packets are included in DCU statistics. If an output filter drops traffic on other M-series platforms, the dropped packets are excluded from DCU statistics.

On T-series and M320 platforms, the `destination-class` and `source-class` statements are not supported at the `[edit firewall family family-name filter filter-name term term-name from]` hierarchy level. On other M-series platforms, these statements are supported.

Once you enable accounting on an interface, the JUNOS software maintains packet counters for that interface, with separate counters for inet, inet6, and mpls protocol families. You must then configure the source class and destination class attributes in policy action statements, which must be included in forwarding-table export policies.

For a complete discussion about source and destination class accounting profiles, see the *JUNOS Network Management Configuration Guide*. For more information about MPLS, see the *JUNOS MPLS Applications Configuration Guide*.

Examples: Enabling Source Class and Destination Class Usage

Configure DCU and SCU output on one interface:

```
[edit]
interfaces {
  so-6/1/0 {
    unit 0 {
      family inet {
        accounting {
          destination-class-usage;
          source-class-usage {
            output;
          }
        }
      }
    }
  }
}
```

Complete SCU Configuration Source routers A and B use loopback addresses as the prefixes to be monitored. Most of the configuration tasks and actual monitoring occur on transit Router SCU.

The loopback address on Router A contains the origin of the prefix that is to be assigned to source class A on Router SCU. However, no SCU processing happens on this router. Therefore, configure Router A for basic Open Shortest Path First (OSPF) routing and include your loopback interface and interface so-0/0/2 in the OSPF process.

```

Router A    [edit]
            interfaces {
              so-0/0/2 {
                unit 0 {
                  family inet {
                    address 10.255.50.2/24;
                  }
                }
              }
              lo0 {
                unit 0 {
                  family inet {
                    address 10.255.192.10/32;
                  }
                }
              }
            }

            protocols {
              ospf {
                area 0.0.0.0 {
                  interface so-0/0/2.0;
                  interface lo0.0;
                }
              }
            }

```

Router SCU Router SCU handles the bulk of the activity in this example. On Router SCU, enable source class usage on the inbound and outbound interfaces at the [edit interfaces *interface-name* unit *unit-number* family inet accounting] hierarchy level. Make sure you specify the expected traffic: input, output, or, in this case, both.

Next, configure a route filter policy statement that matches the prefixes of the loopback addresses from routers A and B. Include statements in the policy that classify packets from Router A in one group named scu-class-a and packets from Router B in a second class named scu-class-b. Notice the efficient use of a single policy containing multiple terms.

Last, apply the policy to the forwarding table.

```
[edit]
interfaces {
  so-0/0/1 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
            output;
          }
        }
      }
      address 10.255.50.1/24;
    }
  }
  so-0/0/3 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
            output;
          }
        }
      }
      address 10.255.10.3/24;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.6.111/32;
      }
    }
  }
}
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/1.0;
      interface so-0/0/3.0;
    }
  }
}
```

```

routing-options {
  forwarding-table {
    export scu-policy;
  }
}
policy-options {
  policy-statement scu-policy {
    term 0 {
      from {
        route-filter 10.255.192.0/24 orlonger;
      }
      then source-class scu-class-a;
    }
    term 1 {
      from {
        route-filter 10.255.165.0/24 orlonger;
      }
      then source-class scu-class-b;
    }
  }
}

```

Router B Just as Router A provides a source prefix, Router B's loopback address matches the prefix assigned to scu-class-b on Router SCU. Again, no SCU processing happens on this router, so configure Router B for basic OSPF routing and include your loopback interface and interface so-0/0/4 in the OSPF process.

```

interfaces {
  so-0/0/4 {
    unit 0 {
      family inet {
        address 10.255.10.4/24;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.165.226/32;
      }
    }
  }
}
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/4.0;
      interface lo0.0;
    }
  }
}

```

**Enabling Packet
Counting for Layer 3
VPNs**

You can use SCU and DCU to count packets on Layer 3 VPNs. To enable packet counting for Layer 3 VPN implementations at the egress point of the MPLS tunnel, you must configure a virtual loopback tunnel interface (vt) on the PE router, map the virtual routing and forwarding (VRF) instance type to the virtual loopback tunnel interface, and send the traffic received from the VPN out the source class output interface, as shown in the following example:

1. Configure a virtual loopback tunnel interface on a provider edge router equipped with a tunnel PIC:

```
[edit interfaces]
vt-0/3/0 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          input;
        }
      }
    }
  }
}
```

2. Map the VRF instance type to the virtual loopback tunnel interface.

For SCU and DCU to work, you must not include the `vrf-table-label` statement at the `[edit routing-instances instance-name]` hierarchy level.

```
[edit routing-instances]
VPN-A {
  instance-type vrf;
  interface at-2/1/1.0;
  interface vt-0/3/0.0;
  route-distinguisher 10.255.14.225:100;
  vrf-import import-policy-A;
  vrf-export export-policy-A;
  protocols {
    bgp {
      group to-r4 {
        local-address 10.27.253.1;
        peer-as 400;
        neighbor 10.27.253.2;
      }
    }
  }
}
```

3. Send traffic received from the VPN out the source class output interface:

```
[edit interfaces]
at-1/1/0 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          output;
        }
      }
    }
  }
}
```

For more information about VPNs, see the *JUNOS VPNs Configuration Guide*. For more information about virtual loopback tunnel interfaces, see “Configuring Tunnel Interfaces” on page 569.

