

Chapter 4

IPv6 Interface Configuration Guidelines

When configuring IPv6 interfaces, you can configure the interfaces that are currently present in the router (that is, the Physical Interface Card [PICs] that are already installed in the router) as well as interfaces that you might be adding at some future time (that is, PICs that you plan to install). To determine which interfaces are currently installed in the router, use the `show interfaces terse` command from the top-level command-line interface (CLI). If an interface is listed in the output, it is installed in the router. If an interface is not listed, it is not present.

The router software automatically configures the router's management Ethernet interface, `fxp0`, which is an out-of-band management interface, and the internal Ethernet interface, `fxp1`, which connects the Routing Engine to the control board (System Control Board [SCB], System and Switch Board [SSB], Forwarding Engine Board [FEB], or Switching and Forwarding Module [SFM]). The software also automatically configures one loopback interface, `lo0`.

Complete IPv6 Interfaces Configuration Statements

To configure IPv6 interfaces, you include statements at the [edit interfaces] hierarchy level of the configuration:

```
interfaces {
  interface-name {
    disable;
    accounting-profile name;
    description text;
    aggregated-ether-options {
      (flow-control | no-flow-control);
      (loopback | no-loopback);
      minimum-links number;
      source-address-filter {
        mac-address;
      }
      (source-filtering | no-source-filtering);
    }
    aggregated-sonet-options {
      link-speed speed;
      minimum-links number;
    }
  }
}
```

```

atm-options {
  vpi vpi-identifier max-vcs maximum-vcs;
  ilmi;
  e3-options {
    atm-encapsulation (direct | PLCP);
    buildout distance (ft | m);
    framing (g751 | g832);
    loopback (local | remote);
    (payload-scrambler | no-payload-scrambler);
  }
  t3-options {
    atm-encapsulation (direct | PLCP);
    buildout distance (ft | m);
    (cbit-parity | no-cbit-parity);
    loopback (local | remote);
    (payload-scrambler | no-payload-scrambler);
  }
}
clocking clock-source;
dce;
e1-options {
  bert-error-rate rate;
  bert-period seconds;
  fcs (32 | 16);
  framing (g704 | unframed);
  idle-cycle-flag (flags | ones);
  loopback (local | remote);
  start-end-flag (shared | filler);
  timeslots slot-number;
}
e3-options {
  bert-algorithm algorithm;
  bert-error-rate rate;
  bert-period seconds;
  compatibility-mode (digital-link | kentrox) <subrate value>;
  fcs (32 | 16);
  idle-cycle-flag value;
  loopback (local | remote);
  (payload-scrambler | no-payload-scrambler);
  start-end-flag value;
}
encapsulation type;
fastether-options {
  802.3ad aeX;
  (flow-control | no-flow-control);
  (loopback | no-loopback);
  source-address-filter {
    mac-address;
  }
  (source-filtering | no-source-filtering);
}
gigether-options {
  802.3ad aeX;
  (flow-control | no-flow-control);
  (loopback | no-loopback);
  source-address-filter {
    mac-address;
  }
  (source-filtering | no-source-filtering);
}

```

```

hold-time up milliseconds down milliseconds;
keepalives <down-count number> <interval seconds> <up-count number>;
link-mode mode;
lmi {
    lmi-type (ansi | itu);
    n391dte number;
    n392dce number;
    n392dte number;
    n393dce number;
    n393dte number;
    t391dte seconds;
    t392dce seconds;
}
mac mac-address;
mtu bytes;
no-keepalives;
no-traps;
receive-bucket {
    overflow (tag | discard);
    rate percentage;
    threshold number;
}
sonet-options {
    aggregate asX;
    aps {
        advertise-interval milliseconds;
        authentication-key key;
        force;
        hold-time milliseconds;
        lockout;
        neighbor address;
        paired-group group-name;
        protect-circuit group-name;
        request;
        revert-time seconds;
        working-circuit group-name;
    }
    bytes {
        e1-quiet value;
        f1 value;
        f2 value;
        s1 value;
        z3 value;
        z4 value;
    }
    fcs (32 | 16);
    loopback (local | remote);
    path-trace trace-string;
    (payload-scrambler | no-payload-scrambler);
    rfc-2615;
    (z0-increment | no-z0-increment);
}

```

```

speed (10m | 100m);
t1-options {
    bert-error-rate rate;
    bert-period seconds;
    buildout (0-133 | 133-266 | 266-399 | 399-532 | 532-655);
    byte-encoding (nx64 | nx56);
    fcs (32 | 16);
    framing (sf | esf);
    idle-cycle-flag (flags | ones);
    invert-data;
    line-encoding (ami | b8zs);
    loopback (local | remote);
    start-end-flag (shared | filler);
    timeslots slot-number;
}
t3-options {
    bert-algorithm algorithm;
    bert-error-rate rate;
    bert-period seconds;
    (cbit-parity | no-cbit-parity);
    compatibility-mode (digital-link | kentrox | larscom) <subrate value>;
    fcs (32 | 16);
    (feac-loop-respond | no-feac-loop-respond);
    idle-cycle-flag value;
    (long-buildout | no-long-buildout);
    loopback (local | remote);
    (payload-scrambler | no-payload-scrambler);
    start-end-flag value;
}
traceoptions {
    flag flag <flag-modifier> <disable>;
}
transmit-bucket {
    overflow (tag | discard);
    rate percentage;
    threshold number;
}
vlan-tagging;
unit logical-unit-number {
    accounting-profile name;
    allow_any_vci;
    disable;
    dlcidlcid-identifier;
    drop-timeout milliseconds;
    encapsulation type;
    family family {
        mtu bytes;
        address address {
            destination destination-address;
            eui-64;
            preferred;
            primary;
        }
    }
}
fragment-threshold bytes;
mrru bytes;
multicast-dlcidlcid-identifier;
multicast-vpivpi-identifier.vci-identifier;
no-traps;
oam-liveness {
    up-count cells;
    down-count cells;
}

```

```

oam-period (disable | seconds);
point-to-point;
shaping {
    (cbr rate | vbr peak rate sustained rate burst length);
    queue-length number;
}
short-sequence;
tunnel {
    source source-address;
    destination destination-address;
    ttl number;
}
vci vpi-identifier.vci-identifier;
vlan-id number;
}
}

```

This chapter describes the following tasks for configuring and monitoring an IPv6 interface:

Minimum IPv6 Interface Configuration on page 39

Configure the Interface Name on page 40

Add an Interface Description to the Configuration on page 41

Specify the Logical Interface Number on page 41

Configure the IPv6 Family on the Interface on page 42

Configure the MTU on page 44

Interface CLI Commands on page 45

System CLI Commands on page 46

Minimum IPv6 Interface Configuration

For your router to function properly, you must configure each PIC interface that is present in the router. No PIC interfaces are preconfigured. This chapter focuses on configuring an interface to support the IPv6 address family. For a complete discussion on configuring interfaces, see the *JUNOS Internet Software Configuration Guide: Interfaces and Class of Service*.

Configure the Interface Name

Each interface has a name that identifies the physical interface type and the location of the interface card in the chassis. To configure the interface name, specify it at the [edit interfaces] hierarchy level:

```
[edit]
interfaces {
  interface-name {
    ...
  }
}
```

Specify the interface name in the following format:

physical<:channel>.logical

The physical part of an interface name has the following format:

type-fpc/pic/port

type is the media type and can be one of the following:

ae—Aggregated Ethernet interface. This is actually a virtual aggregated link and has a different naming format.

as—Aggregated SONET/SDH interface. This is actually a virtual aggregated link and has a different naming format.

at—Asynchronous Transfer Mode (ATM) interface.

e1—E1 interface. If the name does not include a channel identifier, it is assumed to be a copper-cable-based E1 interface. If the name includes a channel identifier, it is assumed to be an STM-1 channel on a Channelized STM-1 to E1 interface.

e3—E3 interface.

fe—Fast Ethernet interface.

fxp—Management and internal Ethernet interfaces.

ge—Gigabit Ethernet interface.

gr—Generic Routing Encapsulation (GRE) tunnel interface.

ip—IP-over-IP encapsulation tunnel interface. This interface type is useful for the configured tunnel, which is discussed later in the manual.

lo—Loopback interface.

ml—Multilink interface.

so—SONET interface.

t1—T1 interface. If the name does not include a channel identifier, it is assumed to be a copper-cable-based T1 interface. If the name includes a channel identifier, it is assumed to be a DS-1 channel on a Channelized DS-3 interface or a Channelized OC-3 to T1 interface.

t3—T3 interface. If the name does not include a channel identifier, it is assumed to be a copper-cable-based T3 interface. If the name includes a channel identifier, it is assumed to be a DS-3 channel on a Channelized OC-12 interface.

fpc is the slot in which the Flexible PIC Concentrator (FPC) card is installed.

pic is the number of the PIC location in which the interface card is installed on the FPC.

port is the specific port on a PIC.

The logical unit part of the interface name corresponds to the logical unit number, which can range from 0 through 65,535.

Add an Interface Description to the Configuration

You can include a text description of each physical interface in the configuration file. Any descriptive text you include is displayed in the output of the `show interfaces` commands. It has no impact on the interface configuration but can be useful in identifying the role of an interface in a network topology. To add a text description, include the description statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]  
description text;
```

The description can be a single line of text. If the text contains spaces, enclose it in quotation marks.

Specify the Logical Interface Number

Each logical interface must have a logical unit number. The logical unit number corresponds to the logical unit part of the interface name.

The Point-to-Point Protocol (PPP) and Cisco High-level Data Link Control (HDLC) encapsulations support only a single logical interface, whose logical unit number must be 0. Frame Relay and ATM encapsulations support multiple logical interfaces, so you can configure one or more logical unit numbers.

You specify the logical unit number in the unit statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces]
interface-name {
  unit 0 {
    ...
  }
}
interface-name {
  unit logical-unit-number {
    ...
  }
}
```

The logical unit number can range from 0 through 65,535.

Configure the IPv6 Family on the Interface

For the interface to support IPv6, you must configure the interface with inet6 as the family type.

To configure the logical interface's protocol family, include the family statement at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level, specifying inet6 as the family. To configure more than one protocol family on a logical interface, include multiple family statements. You can configure IPv6 addressing as well as other supported families (inet, iso, mlppp, mpls, and tnp) on all interface types that support multiple address families.

```
[edit interfaces interface-name unit logical-unit-number]
family inet6 {
  mtu size;
  address address {
    destination address;
    eui-64;
    primary;
    preferred;
  }
}
```

Configure the IPv6 Address on an Interface

You assign an 128-bit IPv6 address to an interface by specifying the address at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6] hierarchy level.

To assign an IPv6 address to an interface, include the address statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6] hierarchy level:

```
[edit interfaces interface-name unit logical-unit-number family inet6]
address address {
  destination address;
  eui-64;
  primary;
  preferred;
}
```

Configure the Destination Address for an Interface

The *destination address* on an interface is the address of the remote side of the connection (for point-to-point interfaces only).

To set a destination address for the subnet, include the destination statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6 address *address*] hierarchy level:

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

Using the Interface Identifier for Non-Link-Local Addresses

The router generates a link-local address for each interface which has family inet6 configured on it. This link-local address contains a 64-bit interface identifier. When configuring a non-link-local address on an interface, you can incorporate this interface identifier into the final address.

To use the interface identifier in the address, include the eui-64 statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6 address *address*] hierarchy level:

```
[edit interfaces interface-name unit logical-unit-number family inet6 address address]
eui-64;
```



The configured IPv6 address must be 64 bits or fewer. The configuration will not commit if an interface identifier has been manually configured already.

Configure 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 packets sourced locally and sent out the interface. By default, the primary address on an interface is selected as the numerically lowest local address configured on the interface.

To set a different IPv6 primary address, include the primary statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6 address *address*] hierarchy level:

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

Configure the Preferred Address for an Interface

The *preferred address* on an interface is the default local address used for packets sourced by the local router to destinations on the subnet. By default, the numerically lowest local address is chosen.

To set a different preferred address for the subnet, include the preferred statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet6 address *address*] hierarchy level:

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

Configure the MTU

For each interface, you can configure an interface-specific maximum transfer unit (MTU) by including the mtu statement at the [edit interfaces interface *interface-name*] hierarchy level. To modify this MTU for the IPv6 protocol family, include the mtu statement at the [edit interfaces interface *interface-name* unit *logical-unit-number* family inet6] hierarchy level:

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

The default MTU for Ethernet encapsulation on IPv6 and IPv4 is 1500 bytes. For Ethernet encapsulation on ISO (IS-IS), the default protocol MTU is 1497 bytes.



Note

When you initially configure an interface, the protocol MTU is calculated automatically. However, if you subsequently change the media MTU, the MTU on existing address families does not automatically adjust.

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. 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.



Note

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 specified as 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.

Example: Configure an IPv6 Address

Configure an IPv6 address on a Fast Ethernet interface:

```

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

```

Interface CLI Commands

From the CLI, you can run show commands to display interface information.

show interfaces terse

Syntax show interfaces terse <interface-name>

Description Display summary information about interfaces.

Options interface-name—(Optional) Name of an interface.

Required Privilege Level view

Output Fields Interface—Name of the interface.

Admin—Whether the interface is turned on (up) or off (down).

Link—Link state. It can be up or down.

Proto—Protocol configured on that interface.

Local—Local address of the interface.

Remote—For point-to-point interfaces, address of the remote side of the connection.

Sample Output

```

user@host> show interfaces terse
Interface      Admin Link Proto  Local          Remote
at-0/0/0       up   up
at-0/0/0       up   up
at-0/0/0       up   down
at-0/0/0       up   down
t3-0/2/0       up   up
t3-0/2/0       up   up   inet   10.21.1.2/30
               inet6   8021::3:0:1/96
               fe80::a0:a5ff:fe3d:dbf/64
ge-1/1/0.0     up   up   inet6   8021::101:0:1/96
               8021::102:0:1/96
               8021::103:0:1/96
               fe80::90:69ff:fe4a:289d/64

```

```

fxp0          up    up
fxp0.0       up    up    inet    192.168.71.1/24
fxp1         up    up
fxp1.0       up    up    tnp     4
fxp2         up    up
fxp2.0       up    up    tnp     4
gre          up    up
ipip         up    up
lo0          up    up
lo0.0        up    up    inet    10.225.71.1      --> 0/0
              127.0.0.1      --> 0/0
pimd         up    up
pime         up    up
tap          up    up

```

System CLI Commands

From the CLI, you can run show commands for system monitoring.

show system connections

Syntax show system connections <extensive> (inet | inet6)

Description Display information about the active IP sockets on the Routing Engine. Use this command to verify which servers are active on a system and what connections are currently in progress.

Options none—Information about the active IP sockets on the Routing Engine. This option is equivalent to the UNIX netstat -an -f inet command.

extensive—(Optional) Display exhaustive system process information, which, for TCP connections, includes the TCP control block. This option is useful for debugging TCP connections. This option is equivalent to the UNIX netstat -Van -f inet command.

inet—(Optional) Display IPv4 connections.

inet6—(Optional) Display IPv6 connections.

Required Privilege Level view

Sample Output Sample Output: show system connections on page 47
Sample Output: show system connections extensive on page 47

Output Fields Proto—Protocol of the socket. It can be either TCP or UDP.

Recv-Q—Number of input packets received by the protocol and waiting to be processed by the application.

Send-Q—Number of output packets sent by the application and waiting to be processed by the protocol.

Local Address—Local address and port of the socket, separated by a period. An asterisk (*) indicates that the bound address is the wildcard address. Server sockets typically have the wildcard address and a well-known port bound to them.

Foreign Address—Foreign address and port of the socket, separated by a period. An asterisk (*) indicates that the address or port is a wildcard.

(state)—For TCP, the protocol state of the socket.

Sample Output: show system connections

```
user@host> show system connections
Active Internet connections (including servers)
Proto Recv-Q Send-Q Local Address          Foreign Address         (state)
tcp      0      2 192.168.4.16.513      208.197.169.254.894   ESTABLISHED
tcp      0      0 192.168.4.16.513      208.197.169.195.945   ESTABLISHED
tcp      0      0 *.23                   *.*                     LISTEN
tcp      0      0 *.22                   *.*                     LISTEN
tcp      0      0 *.513                  *.*                     LISTEN
tcp      0      0 *.514                  *.*                     LISTEN
tcp      0      0 *.21                   *.*                     LISTEN
tcp      0      0 *.79                   *.*                     LISTEN
tcp      0      0 *.1023                 *.*                     LISTEN
tcp      0      0 *.111                  *.*                     LISTEN
udp      0      0 192.168.4.16.1634     208.197.169.249.2049
udp      0      0 192.168.4.16.1627     208.197.169.254.2049
udp      0      0 192.168.4.16.1371     208.197.169.195.2049
udp      0      0 *.*                    *.*                     LISTEN
udp      0      0 *.9999                 *.*                     LISTEN
udp      0      0 *.161                  *.*                     LISTEN
udp      0      0 192.168.4.16.1039     192.168.4.16.1023
udp      0      0 192.168.4.16.1038     192.168.4.16.1023
udp      0      0 192.168.4.16.1037     192.168.4.16.1023
udp      0      0 192.168.4.16.1036     192.168.4.16.1023
udp      0      0 *.1022                 *.*                     LISTEN
udp      0      0 *.1023                 *.*                     LISTEN
udp      0      0 *.111                  *.*                     LISTEN
udp      0      0 *.*                    *.*                     LISTEN
```

Sample Output: show system connections extensive

```
user@host> show system connections extensive
Active Internet connections (including servers)
Proto Recv-Q Send-Q Local Address          Foreign Address         (state)
tcp      0      2 192.168.4.16.513      208.197.169.254.894   ESTABLISHED
      iss: 3972677059      sndup: 3972693435      sndcc: 10
      snduna: 3972693435      sndnxt: 3972693437      sndwnd: 17376
      sndmax: 3972693437      sndcwnd: 65535      sndssthresh: 1073725440
      irs: 484187869      rcvup: 484188060      rcvcc: 98357
      rcvnxt: 484188070      rcvadv: 484205446      rcvwnd: 17376
      rtt: 1      srtt: 7      rttv: 5
      rttmin: 2      duration: 5011      mss: 1448
      flags: REQ_SCALE RCVD_SCALE REQ_TSTMP RCVD_TSTMP RCVD_CC [0x41e0]
tcp      0      0 192.168.4.16.513      208.197.169.195.945   ESTABLISHED
      iss: 1057609890      sndup: 1057790796      sndcc: 2
      snduna: 1057790810      sndnxt: 1057790810      sndwnd: 17376
      sndmax: 1057790810      sndcwnd: 39096      sndssthresh: 1073725440
      irs: 3551947312      rcvup: 3551947422      rcvcc: 0
      rcvnxt: 3551947422      rcvadv: 3551964798      rcvwnd: 17376
      rtt: 0      srtt: 17      rttv: 11
      rttmin: 2      duration: 125814      mss: 1448
      flags: REQ_SCALE RCVD_SCALE REQ_TSTMP RCVD_TSTMP [0x1e0]
udp      0      0 192.168.4.16.1634     208.197.169.249.2049
udp      0      0 192.168.4.16.1627     208.197.169.254.2049
udp      0      0 192.168.4.16.1371     208.197.169.195.2049
udp      0      0 *.*                    *.*                     LISTEN
```



```

Sample Output user@host> show system statistics
ip:
    476559 total packets received
    0 bad header checksums
    0 with size smaller than minimum
    0 with data size < data length
    0 with header length < data size
    0 with data length < header length
    0 with bad options
    0 with incorrect version number
    0 fragments received
    0 fragments dropped (dup or out of space)
    0 fragments dropped after timeout
    0 packets reassembled ok
    406456 packets for this host
    39565 packets for unknown/unsupported protocol
    354 packets forwarded
    0 packets not forwardable
    0 redirects sent
    342678 packets sent from this host
    5 packets sent with fabricated ip header
    0 output packets dropped due to no bufs, etc.
    0 output packets discarded due to no route
    0 output datagrams fragmented
    0 fragments created
    0 datagrams that can't be fragmented

icmp: ...
tcp: ...
udp: ...
igmp: ...
arp: ...
clnl: ...
esis: ...

```

