

## Chapter 7

# Configuring Schedulers

You use *schedulers* to configure transmission scheduling and rate control parameters. Schedulers define the priority, bandwidth, delay buffer size, rate control status, and RED drop profiles to be applied to a particular class of traffic.

You associate the schedulers with forwarding classes by means of scheduler maps. You can then associate each scheduler map with an interface, thereby configuring the hardware queues, packet schedulers, and RED processes that operate according to this mapping.

A scheduler configuration block specifies the buffer size, bandwidth, and priority for a queue. It also specifies the RED drop profile for packets that fall within specification and out of specification.

Physical interfaces (for example, `t3-0/0/0`, `t3-0/0/0:0`, and `ge-0/0/0`) support scheduling with any encapsulation type pertinent to that physical interface. For a single port, you cannot apply scheduling to the physical interface if you have applied scheduling to one or more of the associated logical interfaces.

Logical interfaces (for example, `t3-0/0/0` unit 0 and `ge-0/0/0` unit 0) support scheduling on data link connection identifiers (DLCIs) or VLANs only.

In the JUNOS software implementation, the term *logical interfaces* generally refers to interfaces you configure by including the `unit` statement at the `[edit interfaces interface-name]` hierarchy level. Logical interfaces have the *.logical* descriptor at the end of the interface name, as in `ge-0/0/0.1` or `t1-0/0/0:0.1`, where the logical unit number is 1.

Although channelized interfaces are generally thought of as logical or virtual, the JUNOS software sees T3, T1, and NxDS0 interfaces within a channelized IQ PIC as physical interfaces. For example, both `t3-0/0/0` and `t3-0/0/0:1` are treated as physical interfaces by the JUNOS software. In contrast, `t3-0/0/0.2` and `t3-0/0/0:1.2` are considered logical interfaces because they have the `.2` at the end of the interface names.

Within the `[edit class-of-service]` hierarchy level, you cannot use the *.logical* descriptor when you assign properties to logical interfaces. Instead, you must include the `unit` statement in the configuration. For example:

```
[edit class-of-service]
user@host# set interfaces t3-0/0/0 unit 0 scheduler-map map1
```

To configure class-of-service (CoS) schedulers, you can include the following statements at the [edit class-of-service] hierarchy level of the configuration:

```

class-of-service {
  interfaces {
    interface-name {
      scheduler-map map-name;
      scheduler-map-chassis map-name;
      unit logical-unit-number {
        scheduler-map map-name;
        shaping-rate rate;
      }
    }
  }
  fabric {
    scheduler-map {
      priority (high | low) scheduler scheduler-name;
    }
  }
  scheduler-maps {
    map-name {
      forwarding-class class-name scheduler scheduler-name;
    }
  }
  schedulers {
    scheduler-name {
      buffer-size (percent percentage | remainder | temporal microseconds);
      drop-profile-map loss-priority (any | high | medium | low)
        protocol (any | non-tcp | tcp) drop-profile profile-name;
      priority priority-level;
      transmit-rate (rate | percent percentage | remainder) <exact>;
    }
  }
}

```

This chapter discusses the following topics:

- Default Scheduler on page 61
- Configuring the Scheduler Buffer Size on page 62
- Configuring the Scheduler Drop Profile on page 68
- Configuring Priority Scheduling on page 68
- Configuring the Scheduler Transmission Rate on page 70
- Configuring the Scheduler Map on page 71
- Associating the Scheduler Map with an Output Interface on page 71
- Scheduling Packet Forwarding Component Queues on page 72
- Associating a Scheduler Map with a DLCI or VLAN on page 77
- Associating a Scheduler with a Fabric Priority on page 82

## Default Scheduler

---

The following default scheduler is provided when you install the JUNOS software. These settings are not visible in the output of the `show class-of-service` command; rather, they are implicit.

```
[edit class-of-service]
schedulers {
  network-control {
    transmit-rate percent 5;
    buffer-size percent 5;
    priority low;
    drop-profile-map loss-priority any protocol any drop-profile terminal;
  }
  best-effort {
    transmit-rate percent 95;
    buffer-size percent 95;
    priority low;
    drop-profile-map loss-priority any protocol any drop-profile terminal;
  }
}
drop-profiles {
  terminal {
    fill-level 100 drop-probability 100;
  }
}
```

## Configuring the Scheduler Buffer Size

To control congestion at the output stage, you can configure the delay-buffer bandwidth. The delay-buffer bandwidth provides packet buffer space to absorb burst traffic up to the specified duration of delay. Once the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the head of the buffer.

By default, the buffer sizes for queues 0 through 7 are 95, 0, 0, 5, 0, 0, 0, and 0 percent of the total available buffer space.

To configure the buffer size, include the `buffer-size` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
  buffer-size (percent percentage | remainder | temporal microseconds);
```

For each scheduler, you can configure the buffer size as one of the following:

- A percentage of the total buffer
- The remaining buffer available. The remainder is the buffer percentage that is not assigned to other queues. For example, if you assign 40 percent of the delay buffer to queue 0, allow queue 3 to keep the default allotment of 5 percent, and assign the remainder to queue 7, then queue 7 uses approximately 55 percent of the delay buffer.
- A temporal value, in microseconds. For the temporal setting, the queuing algorithm starts dropping packets when it queues more than a computed number of bytes. This maximum is computed by multiplying the logical interface speed by the configured temporal value. The buffer size temporal value differs by platform type, as shown in Table 14.

For information about configuring large buffer sizes on IQ PICs, see “Configuring Large Delay Buffers for Slower Interfaces” on page 63.

**Table 13: Buffer Size Temporal Value Ranges by Platform Type**

Platforms	Temporal Value Ranges
T-series and M320	1 through 50,000 microseconds
Other M-series	1 through 200,000 microseconds
IQ PICs on all platforms	1 through 100,000 microseconds
<b>With Large Buffer Sizes Enabled</b>	
IQ PICs on all platforms	1 through 500,000 microseconds
Gigabit Ethernet IQ VLANs	
With shaping rate up to 10 mbps	1 through 400,000 microseconds
With shaping rate up to 20 mbps	1 through 300,000 microseconds
With shaping rate up to 30 mbps	1 through 200,000 microseconds
With shaping rate up to 40 mbps	1 through 150,000 microseconds

## Configuring Large Delay Buffers for Slower Interfaces

By default, T1, E1, and NxDS0 interfaces and DLCIs configured on channelized IQ PICs and Gigabit Ethernet VLANs configured on Gigabit Ethernet IQ PICs are limited to 100,000 microseconds of delay buffer. For these interfaces, it might be necessary to configure a larger buffer size to prevent congestion and packet dropping. You can do so on the following PICs:

- 10-port Channelized E1 IQ
- Channelized T3 IQ
- Channelized STM-1
- Gigabit Ethernet IQ

Congestion and packet dropping occur when large bursts of traffic are received by slower interfaces. This happens when faster interfaces pass traffic to slower interfaces, which is often the case when edge devices receive traffic from the core of the network. For example, a 100,000-microsecond T1 delay buffer can absorb only 20 percent of a 5000-microsecond burst of traffic from an upstream OC3 interface. In this case, 80 percent of the burst traffic is dropped.

Table 14 shows some recommended buffer sizes needed to absorb typical burst sizes from various upstream interface types.

**Table 14: Recommended Delay Buffer Sizes**

Length of Burst	Upstream Interface	Downstream Interface	Recommended Buffer on Downstream Interface
5000 microseconds	OC3	E1 or T1	500,000 microseconds
5000 microseconds	E1, T1	E1 or T1	100,000 microseconds
1000 microseconds	T3	E1 or T1	100,000 microseconds

To ensure that traffic is queued and transmitted properly on Gigabit Ethernet VLANs, and E1, T1, and NxDS0 interfaces and DLCIs, you can configure a buffer size larger than the default maximum. To enable larger buffer sizes to be configured, include the `q-pic-large-buffer` statement at the `[edit chassis fpc slot-number pic pic-number]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number]
q-pic-large-buffer;
```

When you include the `q-pic-large-buffer` statement in the configuration, the larger buffer is transparently available for allocation to scheduler queues. The larger buffer maximum varies by interface type, as shown in Table 15 on page 64.

**Table 15: Maximum Delay Buffer with 'q-pic-large-buffer' Enabled by Interface Type**

Interface Types	Maximum Delay Buffer with 'q-pic-large-buffer'
E1 and T1	500,000 microseconds
NxDSO:	
1xDSO through 3xDSO	4,000,000 microseconds
4xDSO through 7xDSO	2,000,000 microseconds
8xDSO through 15xDSO	1,000,000 microseconds
16xDSO through 32xDSO	500,000 microseconds
Gigabit Ethernet IQ VLANs:	
With shaping rate up to 10 mbps	400,000 microseconds
With shaping rate up to 20 mbps	300,000 microseconds
With shaping rate up to 30 mbps	200,000 microseconds
With shaping rate up to 40 mbps	150,000 microseconds

If you configure a delay buffer larger than the new maximum, the candidate configuration can be committed successfully. However, the setting is rejected by the packet forwarding component, the default setting is used instead, and a system log warning message is generated.

For interfaces that support DLCI queuing, the large buffer is supported for DLCIs on which the configured shaping rate is less than or equal to the physical interface bandwidth. For instance, when you configure a Frame Relay DLCI on a Channelized T3 IQ PIC, and you configure the shaping rate to be 1.5 Mbps, the amount of delay buffer that can be allocated to the DLCI is 500,000 microseconds, which is equivalent to a T1 delay buffer. For more information about DLCI queuing, see “Associating a Scheduler Map with a DLCI or VLAN” on page 77.

The larger buffer sizes are enabled for Gigabit Ethernet IQ VLANs but not for physical Gigabit Ethernet interfaces. Therefore, the limitations for physical Gigabit Ethernet IQ interfaces remain as shown in Table 13 on page 62 (50,000 microseconds on T-series and M320 platforms and 200,000 microseconds on other M-series platforms).

For NxDSO interfaces, the larger buffer sizes can be up to 4,000,000 microseconds, depending on the number of DSO channels in the NxDSO interface. For slower NxDSO interfaces with fewer channels, the delay buffer can be relatively larger than for faster NxDSO interfaces with more channels. This is shown in Table 15 on page 64. To calculate specific buffer sizes for various NxDSO interfaces, see “Maximum Delay Buffer for NxDSO Interfaces” on page 65.

You can allocate the delay buffer as either a percentage or a temporal value. The resulting delay buffer is calculated differently depending how you configure the delay buffer, as shown in Table 16 on page 65.

**Table 16: Delay Buffer Calculations**

Delay Buffer Configuration	Formula	Example
Percentage	available interface bandwidth * configured percentage buffer-size * maximum buffer = queue buffer	If you configure a queue on a T1 interface to use 30 percent of the available delay buffer, the queue receives 28,125 bytes of delay buffer:  <pre> sched-expedited {   transmit-rate percent 30;   buffer-size percent 30; } </pre> 1.5 Mbps * 0.3 * 500,000 microseconds = 225,000 bits = 28,125 bytes
Temporal	available interface bandwidth * configured percentage transmit-rate * configured temporal buffer-size = queue buffer	If you configure a queue on a T1 interface to use 500,000 microseconds of delay buffer, and you configure the transmission rate to be 20 percent, the queue receives 18,750 bytes of delay buffer:  <pre> sched-best {   transmit-rate percent 20;   buffer-size temporal 500000; } </pre> 1.5 Mbps * 0.2 * 500,000 microseconds = 150,000 bits = 18,750 bytes
Percentage, with buffer size larger than transmit rate		In this example, the delay buffer is allocated twice the transmit rate. Maximum delay buffer latency can be up to twice the 500,000-microsecond delay buffer if the queue's transmit rate cannot exceed the allocated transmit rate.  <pre> sched-extra-buffer {   transmit-rate percent 10;   buffer-size percent 20; } </pre>

### Maximum Delay Buffer for NxDS0 Interfaces

Because NxDS0 interfaces carry less bandwidth than a T1 or E1 interface, the buffer size on an NxDS0 interface can be relatively larger, depending on the number of DS0 channels combined. The maximum delay buffer size is calculated with the following formula:

$$\text{Interface Speed} * \text{Maximum Delay Buffer Time} = \text{Delay Buffer Size}$$

For example, a 1xDS0 interface has a speed of 64 kilobits (Kb) per second. At this rate, the maximum delay buffer time is 4,000,000 microseconds. Therefore, the delay buffer size is 32 KB:

$$64 \text{ Kb per second} * 4,000,000 \text{ microseconds} = 32 \text{ kilobytes (KB)}$$

Table 17 on page 66 shows the delay buffer calculations for NxDS0 interfaces from 1xDS0 through 32xDS0.

**Table 17: NxDSO Transmission Rates and Delay Buffers**

<b>Interface Speed</b>	<b>Delay Buffer Size</b>
<b>1xDSO Through 4xDSO: Maximum Delay Buffer Time Is 4,000,000 Microseconds</b>	
1xDSO: 64 Kb per second	32 KB
2xDSO: 128 Kb per second	64 KB
3xDSO: 192 Kb per second	96 KB
<b>4xDSO Through 7xDSO: Maximum Delay Buffer Time Is 2,000,000 Microseconds</b>	
4xDSO: 256 Kb per second	64 KB
5xDSO: 320 Kb per second	80 KB
6xDSO: 384 Kb per second	96 KB
7xDSO: 448 Kb per second	112 KB
<b>8xDSO Through 15xDSO: Maximum Delay Buffer Time Is 1,000,000 Microseconds</b>	
8xDSO: 512 Kb per second	64 KB
9xDSO: 576 Kb per second	72 KB
10xDSO: 640 Kb per second	80 KB
11xDSO: 704 Kb per second	88 KB
12xDSO: 768 Kb per second	96 KB
13xDSO: 832 Kb per second	104 KB
14xDSO: 896 Kb per second	112 KB
15xDSO: 960 Kb per second	120 KB
<b>16xDSO Through 32xDSO: Maximum Delay Buffer Time Is 500,000 Microseconds</b>	
16xDSO: 1024 Kb per second	64 KB
17xDSO: 1088 Kb per second	68 KB
18xDSO: 1152 Kb per second	72 KB
19xDSO: 1216 Kb per second	76 KB
20xDSO: 1280 Kb per second	80 KB
21xDSO: 1344 Kb per second	84 KB
22xDSO: 1408 Kb per second	88 KB
23xDSO: 1472 Kb per second	92 KB
24xDSO: 1536 Kb per second	96 KB
25xDSO: 1600 Kb per second	100 KB
26xDSO: 1664 Kb per second	104 KB
27xDSO: 1728 Kb per second	108 KB
28xDSO: 1792 Kb per second	112 KB
29xDSO: 1856 Kb per second	116 KB
30xDSO: 1920 Kb per second	120 KB
31xDSO: 1984 Kb per second	124 KB
32xDSO: 2048 Kb per second	128 KB

**Example: Configuring Large Delay Buffers for Slower Interfaces**

Set large delay buffers on interfaces configured on a Channelized DS3 IQ PIC. The CoS configuration binds a scheduler map to the interface specified in the chassis configuration. For information about the delay buffer calculations in this example, see Table 16 on page 65.

```

chassis {
  fpc 3 {
    pic 0 {
      q-pic-large-buffer;    # ChDS3-IQ in FPC slot 3, PIC slot 0
    }
  }
}

class-of-service {
  interfaces {
    t1-3/0/0:1 {
      scheduler-map large-buf-sched-map;
    }
  }
  scheduler-maps {
    large-buf-sched-map {
      forwarding-class best-effort scheduler sched-best;
      forwarding-class expedited-forwarding scheduler sched-expedited;
      forwarding-class assured-forwarding scheduler sched-assured;
      forwarding-class network-control scheduler sched-network;
    }
  }
  schedulers {
    sched-best {
      transmit-rate percent 20;
      buffer-size temporal 500000;
    }
    sched-expedited {
      transmit-rate percent 30;
      buffer-size percent 30;
    }
    sched-assured {
      transmit-rate percent 40;
      buffer-size percent 40;
    }
    sched-network {
      transmit-rate percent 10;
      buffer-size percent 10;
    }
    sched-extra-buffer {
      transmit-rate percent 10;
      buffer-size percent 20;
    }
  }
}

```

## Configuring the Scheduler Drop Profile

---

The scheduler drop profile defines the drop probabilities across the range of delay-buffer occupancy, thereby supporting the RED process. Depending on the drop probabilities, RED might drop packets aggressively long before the buffer becomes full, or it might drop only a few packets even if the buffer is almost full.

By default, the drop profile is mapped to packets with low PLP, regardless of protocol type. To configure which packet types are mapped to a specified drop profile, include the `drop-profile-map` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
  drop-profile-map loss-priority (any | high | medium | low) protocol (any | non-tcp | tcp)
    drop-profile profile-name;
```

The map sets the drop profile for a specific PLP and protocol type. The inputs for the map are the PLP and the protocol type. The output is the drop profile. For more information about how CoS maps work, see Table 5 on page 14.

For each scheduler, you can configure four separate drop profiles, one for each combination of loss priority (low or high) and IP transport protocol (TCP/IP or non-TCP/IP).

You can configure a maximum of 32 different drop profiles.

For more information, see “Configuring RED Drop Profiles” on page 85.

## Configuring Priority Scheduling

---

Priority scheduling determines the order in which an output interface transmits traffic from the queues. The JUNOS software supports multiple levels of transmission priority, which in order of increasing priority are `low`, `medium-low`, `medium-high`, and `high`.

The JUNOS software priorities map to numeric priorities in the underlying hardware. Two software priorities behave differently only if they map to two distinct hardware priorities. For more information, see “Hardware Priority Mappings” on page 70.

To configure priority scheduling, include the `priority` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
  priority priority-level;
```

Higher-priority queues transmit packets ahead of lower priority queues as long as the higher-priority forwarding classes retain enough bandwidth credit. When you configure a higher-priority queue with a significant fraction of the transmission bandwidth, the queue might lock out lower priority traffic.

You can also configure one queue per interface to have **strict-high** priority, which works the same as **high** priority, but provides unlimited transmission bandwidth. As long as the queue with **strict-high** priority has traffic to send, it receives precedence over all other queues, except queues with **high** priority. Queues with **strict-high** and **high** priority take turns transmitting packets until the **strict-high** queue is empty, the **high** priority queues are empty, or the **high** priority queues run out of bandwidth credit. Only then can lower priority queues send traffic.

The **high** priority allows you to protect traffic classes from being starved by traffic in a **strict-high** queue. For example, a network-control queue might require a small bandwidth allocation (say, 5 percent). You can assign **high** priority to this queue to prevent it from being underserved.



**NOTE:** When you configure a queue to have **strict-high** priority, you should not include the **transmit-rate** statement in the queue configuration because the transmission rate of a **strict-high** priority queue is not limited by the WRR configuration.

The **strict-high** priority works differently on AS PIC link services IQ (lsq) interfaces. For link services IQ interfaces, a **strict-high**-priority queue might starve all the other queues. For more information, see the *JUNOS Services Interfaces Configuration Guide*.

### Example: Configuring Priority Scheduling

Configure priority scheduling, as shown in the following example:

1. Configure a scheduler, **be-sched**, with **medium-low** priority.

```
[edit class-of-service]
schedulers {
  be-sched {
    priority medium-low;
  }
}
```

2. Configure a scheduler map, **be-map**, that associates **be-sched** with the **best-effort** forwarding class.

```
[edit class-of-service]
scheduler-maps {
  be-map {
    forwarding-class best-effort scheduler be-sched;
  }
}
```

3. Assign **be-map** to a Gigabit Ethernet interface, **ge-0/0/0**.

```
[edit class-of-service]
interfaces {
  ge-0/0/0 {
    scheduler-map be-map;
  }
}
```

## Hardware Priority Mappings

The priority levels you configure map to hardware priority levels. These priority mappings depend on the FPC type in which the PIC is mounted.

Table 18 shows the priority mappings by FPC type. Note, for example, that on M320 FPCs and T-series enhanced FPCs, the software priorities `medium-low` and `medium-high` behave similarly because they map to the same hardware priority level.

**Table 18: Scheduling Priority Mappings by FPC Type**

Priority Levels	Mappings for FPCs	Mappings for M320 FPCs and T-series Enhanced FPCs
low	0	0
medium-low	0	1
medium-high	1	1
high	1	2
strict-high (full interface bandwidth)	1	2

## Configuring the Scheduler Transmission Rate

The transmission rate control determines the actual traffic bandwidth from each of the forwarding classes you configure. The rate is specified in bits per second. You can limit the transmission bandwidth to the exact value you configure, or allow it to exceed the configured rate if additional bandwidth is available from other queues.

For 8-port, 12-port, and 48-port Fast Ethernet PICs, transmission scheduling is not supported.

By default, the transmission rates for queues 0 through 7 are 95, 0, 0, 5, 0, 0, 0, and 0 percent. To configure transmission scheduling, include the `transmit-rate` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
  transmit-rate (rate | percent percentage | remainder) <exact>;
```

You can specify the transmit rate as follows:

- *rate*—Transmission rate, in bits per second. The rate can be from 3200 through 32,000,000,000 bits per second (bps).
- *percent percentage*—Percentage of transmission capacity.
- *remainder*—Use remaining rate available. In the configuration, you cannot combine the *remainder* and *exact* options.
- *exact*—Enforce the exact transmission rate or percentage you configure with the *transmit-rate rate* or *transmit-rate percent* statement. Under sustained congestion, a rate-controlled queue that goes into negative credit fills up and eventually drops packets. You specify the *exact* option as follows:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate rate exact;
```

```
[edit class-of-service schedulers scheduler-name]
transmit-rate percent percent exact;
```

In the configuration, you cannot combine the *remainder* and *exact* options.

## Configuring the Scheduler Map

---

Once you define a scheduler, you can include it in a *scheduler map*, which maps a specified forwarding class to a scheduler configuration. To do this, include the *scheduler-maps* statement at the [edit class-of-service] hierarchy level:

```
[edit class-of-service]
scheduler-maps {
  map-name {
    forwarding-class class-name scheduler scheduler-name;
  }
}
```

## Associating the Scheduler Map with an Output Interface

---

After you have defined the scheduler map, as described in “Configuring the Scheduler Map” on page 71, you can associate it with an output interface. To do this, include the *scheduler-map* statement at the [edit class-of-service interfaces *interface-name*] hierarchy level:

```
[edit class-of-service interfaces interface-name]
scheduler-map map-name;
```

Interface wildcards are supported.

Generally, you can associate schedulers with physical interfaces only. For some IQ interfaces, you can also associate schedulers with the logical interface. For more information, see “Associating a Scheduler Map with a DLCI or VLAN” on page 77.



**NOTE:** For original Channelized OC12 PICs, limited CoS functionality is supported. For more information, contact Juniper Networks customer support.

## Scheduling Packet Forwarding Component Queues

On IQ interfaces, the traffic that is fed from the packet forwarding components into the PIC uses low PLP by default and is distributed evenly across the four chassis queues (not PIC queues), regardless of the scheduling configuration for each logical interface. This default behavior can cause traffic congestion.

To control the aggregated traffic transmitted from the chassis queues into the PIC, you can configure the chassis queues to derive their scheduling configuration from the associated logical interface’s. Include the `scheduler-map-chassis derived` statement at the `[edit class-of-service interfaces type-fpc/pic]` hierarchy level:

```
[edit class-of-service interfaces type-fpc/pic/*]
scheduler-map-chassis derived;
```



**CAUTION:** If you include the `scheduler-map-chassis derived` statement in the configuration, packet loss might occur when you subsequently add or remove logical interfaces at the `[edit interfaces interface-name]` hierarchy level.

You can include both the `scheduler-map` and the `scheduler-map-chassis derived` statements in the same interface configuration. The `scheduler-map` statement controls the scheduler inside the PIC, while the `scheduler-map-chassis derived` statement controls the aggregated traffic transmitted into the entire PIC. For the Gigabit Ethernet IQ PIC, include both statements.

For more information about the `scheduler-map` statement, see “Associating the Scheduler Map with an Output Interface” on page 71. For information about logical interface scheduling configuration, see “Associating a Scheduler Map with a DLCI or VLAN” on page 77.

Generally, when you include the `scheduler-map-chassis` statement in the configuration, you must use an interface wildcard for the interface name, as in `type-fpc/pic/*`. The wildcard must use this format—for example, `so-1/2/*`, which means all interfaces on FPC slot 1, PIC slot 2. There is one exception—you can apply the chassis scheduler map to a specific interface on the Gigabit Ethernet IQ PIC only.

According to JUNOS software wildcard rules, specific interface configurations override wildcard configurations. For chassis scheduler map configuration, this rule does not apply; instead, specific interface CoS configurations are added to the chassis scheduler map configuration. For more information about how wildcards work with chassis scheduler maps, see “Examples: Scheduling Packet Forwarding Component Queues” on page 73. For general information about wildcards, see the *JUNOS System Basics Configuration Guide*.

For more details, see:

- Assigning a Custom Scheduler to the Packet Forwarding Component Queues on page 73
- Examples: Scheduling Packet Forwarding Component Queues on page 73

### **Assigning a Custom Scheduler to the Packet Forwarding Component Queues**

Optionally, you can apply a custom scheduler to the chassis queues instead of configuring the chassis queues to automatically derive their scheduling configuration from the logical interfaces on the PIC.

To assign a custom scheduler to the packet forwarding component queues, include the `scheduler-map-chassis` statement at the [edit class-of-service interfaces *type-fpc/pic*] hierarchy level:

```
[edit class-of-service interfaces type-fpc/pic/*]
scheduler-map-chassis map-name;
```

For information about defining the scheduler map referenced by *map-name*, see “Configuring the Scheduler Map” on page 71.

### **Examples: Scheduling Packet Forwarding Component Queues**

#### **Associating a Chassis Scheduler Map with a 2-Port IQ PIC**

The two interfaces are so-0/1/0 and so-0/1/1.

According to customary wildcard rules, the so-0/1/0 configuration overrides the so-0/1/\* configuration, implying that the chassis scheduler map MAP1 is not applied to so-0/1/0. However, the wildcard rule is not obeyed in this case; the chassis scheduler map applies to both interfaces so-0/1/0 and so-0/1/1.

```
class-of-service {
  interfaces {
    so-0/1/0 {
      unit 0 {
        classifiers {
          inet-precedence default;
        }
      }
    }
    so-0/1/* {
      scheduler-map-chassis derived;
    }
  }
}
```

**Not Recommended: Gigabit Ethernet IQ Example** On a Gigabit Ethernet IQ PIC, you can apply the chassis scheduler map at both the specific interface level and the wildcard level. We do not recommend this because the wildcard chassis scheduler map takes precedence, which might not be the desired effect. For example, if you want to apply the chassis scheduler map MAP1 to port 0 and MAP2 to port 1, the following is not recommended.

```
[edit class-of-service]
interfaces {
  ge-0/1/0 {
    scheduler-map-chassis MAP1;
  }
  ge-0/1/* {
    scheduler-map-chassis MAP2;
  }
}
```

**Recommended: Gigabit Ethernet IQ Example** Instead, we recommend this:

```
[edit class-of-service]
interfaces {
  ge-0/1/0 {
    scheduler-map-chassis MAP1;
  }
  ge-0/1/1 {
    scheduler-map-chassis MAP2;
  }
}
```

**Configuring ATM CoS with a Normal Scheduler and a Chassis Scheduler** For ATM2 IQ interfaces, the CoS configuration differs significantly from that of other interface types. For more information about ATM CoS, see “Configuring CoS on ATM Interfaces” on page 133.

```
[edit class-of-service]
interfaces {
  at-1/2/* {
    scheduler-map-chassis derived;
  }
}

[edit interfaces]
at-1/2/0 {
  atm-options {
    vpi 0;
    linear-red-profiles red-profile-1 {
      queue-depth 35000 high-plp-threshold 75 low-plp-threshold 25;
    }
  }
  scheduler-maps map-1 {
    vc-cos-mode strict;
    forwarding-class best-effort {
      priority low;
      transmit-weight percent 25;
      linear-red-profile red-profile-1;
    }
  }
}
```

```

unit 0 {
  vci 0.128;
  shaping {
    vbr peak 20m sustained 10m burst 20;
  }
  atm-scheduler-map map-1;
  family inet {
    address 192.168.0.100/32 {
      destination 192.168.0.101;
    }
  }
}
}

```

**Configuring Two T3  
Interfaces on a  
Channelized DS3 IQ PIC**

```

[edit interfaces]
ct3-3/0/0 {
  no-partition interface-type t3; # use entire port 0 as T3
}
ct3-3/0/1 {
  no-partition interface-type t3; # use entire port 1 as T3
}
t3-3/0/0 {
  unit 0 {
    family inet {
      address 10.0.100.1/30;
    }
  }
}
t3-3/0/1 {
  unit 0 {
    family inet {
      address 10.0.101.1/30;
    }
  }
}
}

```

**Associating Normal  
Schedulers with the  
Two T3 Interfaces**

Configure a scheduler for the aggregated traffic transmitted into both T3 interfaces.

```

[edit class-of-service]
interfaces {
  t3-3/0/0 {
    scheduler-map sched-qct3-0;
  }
  t3-3/0/1 {
    scheduler-map sched-qct3-1;
  }
}
}

```

```

scheduler-maps {
  sched-qct3-0 {
    forwarding-class best-effort scheduler be-qct3-0;
    forwarding-class expedited-forwarding scheduler ef-qct3-0;
    forwarding-class assured-forwarding scheduler as-qct3-0;
    forwarding-class network-control scheduler nc-qct3-0;
  }
  sched-qct3-1 {
    forwarding-class best-effort scheduler be-qct3-1;
    forwarding-class expedited-forwarding scheduler ef-qct3-1;
    forwarding-class assured-forwarding scheduler as-qct3-1;
    forwarding-class network-control scheduler nc-qct3-1;
  }
  sched-chassis-to-q {
    forwarding-class best-effort scheduler be-chassis;
    forwarding-class expedited-forwarding scheduler ef-chassis;
    forwarding-class assured-forwarding scheduler as-chassis;
    forwarding-class network-control scheduler nc-chassis;
  }
}
schedulers {
  be-qct3-0 {
    transmit-rate percent 40;
  }
  ef-qct3-0 {
    transmit-rate percent 30;
  }
  as-qct3-0 {
    transmit-rate percent 20;
  }
  nc-qct3-0 {
    transmit-rate percent 10;
  }
  ...
}

```

**Associating a Chassis Scheduler with the Two T3 Interfaces**

Bind a scheduler to the aggregated traffic transmitted into the entire PIC. The chassis scheduler controls the traffic from the packet forwarding components feeding the interface t3-3/0/\*.

```

[edit class-of-service]
interfaces {
  t3-3/0/* {
    scheduler-map-chassis derived;
  }
}

```

## Associating a Scheduler Map with a DLCI or VLAN

---

*Logical interface scheduling* allows you to associate schedulers with interfaces you configure at the [edit interfaces *interface-name* unit *logical-unit-number*] hierarchy level.

By default, transmission scheduling is not enabled on logical interfaces. You can enable logical interface transmission scheduling on the following PICs:

- Channelized E1 IQ PIC
- Channelized OC3 IQ PIC
- Channelized OC12 IQ PIC
- Channelized STM1 IQ PIC
- Channelized T3 IQ PIC
- E3 IQ PIC
- Gigabit Ethernet IQ PIC

On J-series Services Routers only, you can configure per-unit scheduling for virtual channels. For more information, see “Configuring Virtual Channels” on page 121.

Physical interfaces (for example, `t3-0/0/0`, `t3-0/0/0:0`, and `ge-0/0/0`) support scheduling with any encapsulation type pertinent to that physical interface. For a single port, you cannot apply scheduling to the physical interface if you apply scheduling to one or more of the associated logical interfaces.

Logical interfaces (for example, `t3-0/0/0.0`, `ge-0/0/0.0`, and `t1-0/0/0:0.1`) support scheduling on DLCIs or VLANs only. Furthermore, logical interface scheduling is not supported on PICs that do not have IQ.



**NOTE:** In the JUNOS software implementation, the term *logical interfaces* generally refers to interfaces you configure by including the `unit` statement at the [edit interfaces *interface-name*] hierarchy level. As such, logical interfaces have the *.logical* descriptor at the end of the interface name, as in `ge-0/0/0.1` or `t1-0/0/0:0.1`, where the logical unit number is 1.

Although channelized interfaces are generally thought of as logical or virtual, the JUNOS software sees T3, T1, and NxDS0 interfaces within a channelized IQ PIC as physical interfaces. For example, both `t3-0/0/0` and `t3-0/0/0:1` are treated as physical interfaces by the JUNOS software. In contrast, `t3-0/0/0.2` and `t3-0/0/0:1.2` are considered logical interfaces because they have the `.2` at the end of the interface names.

Within the [edit class-of-service] hierarchy level, you cannot use the *.logical* descriptor when you assign properties to logical interfaces. Instead, you must include the `unit` statement in the configuration. For example:

```
[edit class-of-service]
user@host# set interfaces t3-0/0/0 unit 0 scheduler-map map1
```

---

Table 19 shows the interfaces that support transmission scheduling.

**Table 19: Transmission Scheduling Support by Interfaces Type**

Interface Type	Supported	Examples
<b>IQ PICs</b>		
Physical interfaces	Yes	Example of supported configuration, configured on ATM2 IQ PIC: <pre>[edit class-of-service interfaces at-0/0/0] scheduler-map map-1;</pre>
Channelized interfaces configured on IQ PICs	Yes	Example of supported configuration, configured on Channelized DS3 IQ PIC: <pre>[edit class-of-service interfaces t1-0/0/0:1] scheduler-map map-1;</pre>
Logical interfaces (DLCIs and VLANs only) configured on IQ PICs	Yes	Examples of supported configurations: <ul style="list-style-type: none"> <li>■ Configured on Gigabit Ethernet IQ PIC with VLAN tagging enabled:  <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre> </li> <li>■ Configured on E3 IQ PIC with Frame Relay encapsulation:  <pre>[edit class-of-service interfaces e3-0/0/0 unit 1] scheduler-map map-1;</pre> </li> <li>■ Configured on Channelized T3 IQ PIC with Frame Relay encapsulation:  <pre>[edit class-of-service interfaces t1-0/0/0 unit 1] scheduler-map map-1;</pre> </li> </ul>
Logical interfaces configured on IQ PICs (Interfaces that are not DLCIs or VLANs)	No	Example of unsupported configurations: <ul style="list-style-type: none"> <li>■ Configured on E3 IQ PIC with Cisco HDLC encapsulation:  <pre>[edit class-of-service interfaces e3-0/0/0 unit 1] scheduler-map map-1;</pre> </li> <li>■ Configured on ATM2 IQ PIC with LLC/SNAP encapsulation:  <pre>[edit class-of-service interfaces at-0/0/0 unit 1] scheduler-map map-1;</pre> </li> <li>■ Configured on Channelized OC12 IQ PIC with PPP encapsulation:  <pre>[edit class-of-service interfaces t1-0/0/0:1 unit 1] scheduler-map map-1;</pre> </li> </ul>

Interface Type	Supported	Examples
<b>Non-IQ PICs</b>		
Physical interfaces	Yes	Example of supported configuration, configured on T3 PIC: [edit class-of-service interfaces t3-0/0/0] scheduler-map map-1;
Channelized OC12 PIC	Yes	Example supported configuration, configured on Channelized OC12 PIC: [edit class-of-service interfaces t3-0/0/0:1] scheduler-map map-1;
Channelized interfaces (except the Channelized OC12 PIC)	No	Example unsupported configuration, configured on Channelized STM1 PIC: [edit class-of-service interfaces e1-0/0/0:1] scheduler-map map-1;
Logical interfaces	No	Examples of unsupported configurations: <ul style="list-style-type: none"> <li>■ Configured on Fast Ethernet PIC: [edit class-of-service interfaces fe-0/0/0 unit 1] scheduler-map map-1;</li> <li>■ Configured on Gigabit Ethernet PIC: [edit class-of-service interfaces ge-0/0/0 unit 0] scheduler-map map-1;</li> <li>■ Configured on ATM1 PIC: [edit class-of-service interfaces at-0/0/0 unit 2] scheduler-map map-1;</li> <li>■ Configured on Channelized OC12 PIC: [edit class-of-service interfaces t3-0/0/0:0 unit 2] scheduler-map map-1;</li> </ul>

To configure transmission scheduling on logical interfaces, perform the following steps:

1. Enable scheduling on the interface by including the `per-unit-scheduler` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]  
per-unit-scheduler;
```

2. Associate a scheduler with the interface by including the `scheduler-map` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]  
scheduler-map map-name;
```

3. Configure shaping on the interface by including the `shaping-rate` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]  
shaping-rate rate;
```

By default, the logical interface bandwidth is the average of unused bandwidth for the number of logical interfaces that require default bandwidth treatment. You can specify a peak bandwidth rate in bps, either as a complete decimal number or as a decimal number followed by the abbreviation `k` (1000), `m` (1,000,000), or `g` (1,000,000,000). The range is from 1000 through 32,000,000,000 bps.

The sum of the bandwidth you allocate to all logical interfaces on a physical interface must not exceed the bandwidth of the physical interface.

**Example: Associating a Scheduler Map Name with a DLCI or VLAN**

Associate the scheduler `sched-map-logical-0` with logical interface `unit 0` on physical interface `t3-1/0/0`, and allocate 10 megabits per second (Mbps) of transmission bandwidth to the logical interface.

Associate the scheduler `sched-map-logical-1` with logical interface `unit 1` on physical interface `t3-1/0/0`, and allocate 20 Mbps of transmission bandwidth to the logical interface.

The allocated bandwidth is shared among the individual forwarding classes in the scheduler map. Although these schedulers are configured on a single physical interface, they are independent from each other. Traffic on one logical interface unit does not affect the transmission priority, bandwidth allocation, or drop behavior on the other logical interface unit.

```
[edit interfaces]
t3-1/0/0:1 {
  encapsulation frame-relay;
  per-unit-scheduler;
}

[edit class-of-service]
interfaces {
  t3-1/0/0:1 {
    unit 0 {
      scheduler-map sched-map-logical-0;
      shaping-rate 10m;
    }
    unit 1 {
      scheduler-map sched-map-logical-1;
      shaping-rate 20m;
    }
  }
}

scheduler-maps {
  sched-map-logical-0 {
    forwarding-class best-effort scheduler sched-best-effort-0;
    forwarding-class assured-forwarding scheduler sched-bronze-0;
    forwarding-class expedited-forwarding scheduler sched-silver-0;
    forwarding-class network-control scheduler sched-gold-0;
  }
  sched-map-logical-1 {
    forwarding-class best-effort scheduler sched-best-effort-1;
    forwarding-class assured-forwarding scheduler sched-bronze-1;
    forwarding-class expedited-forwarding scheduler sched-silver-1;
    forwarding-class network-control scheduler sched-gold-1;
  }
}
```

```

schedulers {
  sched-best-effort-0 {
    transmit-rate 4m;
  }
  sched-bronze-0 {
    transmit-rate 3m;
  }
  sched-silver-0 {
    transmit-rate 2m;
  }
  sched-gold-0 {
    transmit-rate 1m;
  }
  sched-best-effort-1 {
    transmit-rate 8m;
  }
  sched-bronze-1 {
    transmit-rate 6m;
  }
  sched-silver-1 {
    transmit-rate 4m;
  }
  sched-gold-1 {
    transmit-rate 2m;
  }
}

```

## Associating a Scheduler with a Fabric Priority

---

On M320 and T-series platforms only, you can associate a scheduler with a class of traffic that has a specific priority while transiting the fabric. Traffic transiting the fabric can have two priority values: **low** or **high**. To associate a scheduler with a fabric priority, include the **priority** and **scheduler** statements at the [edit class-of-service fabric scheduler-map] hierarchy level:

```

[edit class-of-service fabric scheduler-map]
priority (high | low) scheduler scheduler-name;

```



**NOTE:** For a scheduler that you associate with a fabric priority, you can include only the **drop-profile-map** statement at the [edit class-of-service schedulers *scheduler-name*] hierarchy level. You cannot include the **buffer-size**, **transmit-rate**, and **priority** statements at that hierarchy level.

---

For information about associating a forwarding class with a fabric priority, see “Overriding Fabric Priority Queuing” on page 36.

**Example: Associating a Scheduler with a Fabric Priority**

Associate a scheduler with a class of traffic that has a specific priority while transiting the fabric:

```
[edit class-of-service]
schedulers {
  fab-be-scheduler {
    drop-profile-map loss-priority low protocol any drop-profile fab-profile-1;
    drop-profile-map loss-priority high protocol any drop-profile fab-profile-2;
  }
  fab-ef-scheduler {
    drop-profile-map loss-priority low protocol any drop-profile fab-profile-3;
    drop-profile-map loss-priority high protocol any drop-profile fab-profile-4;
  }
}
drop-profiles {
  fab-profile-1 {
    fill-level 100 drop-probability 100;
    fill-level 85 drop-probability 50;
  }
  fab-profile-2 {
    fill-level 100 drop-probability 100;
    fill-level 95 drop-probability 50;
  }
  fab-profile-3 {
    fill-level 75 drop-probability 100;
    fill-level 95 drop-probability 50;
  }
  fab-profile-4 {
    fill-level 100 drop-probability 100;
    fill-level 80 drop-probability 50;
  }
}
fabric {
  scheduler-map {
    priority low scheduler fab-be-scheduler;
    priority high scheduler fab-ef-scheduler;
  }
}
```

