

Chapter 29

Configuring SONET/SDH Interfaces

Synchronous Digital Hierarchy (SDH) is a CCITT standard for a hierarchy of optical transmission rates. Synchronous Optical Network (SONET) is a USA standard that is largely equivalent to SDH. Both are widely used methods for very high speed transmission of voice and data signals across the numerous world-wide fiber-optic networks.

SDH and SONET use light-emitting diodes or lasers to transmit a binary stream of light-on and light-off sequences at a constant rate. At the far end optical sensors convert the pulses of light back to electrical representations of the binary information.

In wavelength-division multiplexing (WDM), light at several different wavelengths (colors to a human eye) is transmitted on the same fiber segment, greatly increasing the throughput of each fiber cable.

In dense wavelength-division multiplexing (DWDM), many optical data streams at different wavelengths are combined into one fiber.

The basic building block of the SONET/SDH hierarchy in the optical domain is an OC1; in the electrical domain, it is an STS-1. An OC1 operates at 51.840 Mbps. OC3 operates at 155.520 Mbps.

A SONET/SDH stream can consist of discrete lower-rate traffic flows that have been combined using time-division multiplexing (TDM) techniques. This method is useful, but a portion of the total bandwidth is consumed by the TDM overhead. When a SONET/SDH stream consists of only a single, very high speed payload, it is referred to as operating in concatenated mode. A SONET/SDH interface operating in this mode has a "c" added to the rate descriptor. For example, a concatenated OC48 interface is referred to as OC48c.

SONET and SDH traffic streams exhibit very few differences in behavior that are significant to Juniper Networks SONET/SDH interfaces; in general, this chapter uses *SONET/SDH* to indicate behavior that is identical for the two standards. However, there is one important difference that requires you to configure the interface specifically for SONET or SDH mode. That difference is in the setting of two bits (the ss-bits) in the pointer. SONET equipment ignores these bits, but SDH equipment uses them to distinguish a VC-4 payload from other types. When configured in SDH mode, Juniper Networks SONET/SDH PICs set the ss-bits to s1s0 2 (binary 10). For more information, see the *JUNOS System Basics Configuration Guide*.

This chapter discusses configuration of the following SONET/SDH interface properties:

Configuring SONET/SDH Physical Interface Properties on page 510

Configuring the Media MTU on page 530

Enabling Passive Monitoring on SONET/SDH Interfaces on page 530

Configuring the Clock Source on page 532

Configuring Receive and Transmit Leaky Bucket Properties on page 533

Damping Interface Transitions on page 535

Configuring Interface Encapsulation on page 535

Configuring Aggregated SONET/SDH Interfaces on page 538

For examples of SONET/SDH interface configuration, see the following sections:

Example: Configuring SONET/SDH Interfaces on page 538

Example: Configuring Aggregated SONET/SDH Interfaces on page 543

Configuring SONET/SDH Physical Interface Properties

To configure SONET/SDH physical interface properties, include the `sonet-options` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces so-fpc/pic/port]
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;
    switching-mode (bidirectional | unidirectional);
    working-circuit group-name;
  }
  bytes {
    e1-quiet value;
    f1 value;
    f2 value;
    s1 value;
    z3 value;
    z4 value;
  }
}
```

```

fcs (32 | 16);
loopback (local | remote);
mpls {
  pop-all-labels {
    required-depth number;
  }
}
path-trace trace-string;
(payload-scrambler | no-payload-scrambler);
rfc-2615;
trigger {
  defect ignore {
    hold-time up milliseconds down milliseconds;
  }
}
vtmapping (itu-t | klm);
(z0-increment | no-z0-increment);
}

```

Note that when you configure SONET/SDH OC48 interfaces for channelized (multiplexed) mode (by including the `no-concatenate` statement at the [edit chassis fpc *slot-number* pic *pic-number*] hierarchy level), the bytes f1 statement has no effect. Currently, the bytes e1-quiet statement is ignored if you include it in the configuration. The bytes f2, bytes z3, bytes z4, and path-trace options work correctly on channel 0 and work in the transmit direction only on channels 1, 2, and 3. When using `no-concatenate`, you must specify a channel. For more information, see the *JUNOS System Basics Configuration Guide*.

For DS3 channels on a channelized OC12 interface, the bytes f1, bytes f2, bytes z3, and bytes z4 options have no effect. The bytes s1 option is supported only for channel 0; it is ignored if configured on channels 1 through 11. The bytes s1 value configured on channel 0 applies to all channels on the interface.

You can also include some of the statements in the `sonet-options` statement to set SONET/SDH parameters on ATM interfaces.

You can configure the following SONET/SDH physical interface properties:

Configuring SONET/SDH Header Byte Values on page 512

Configuring an Incrementing STM ID on page 513

Configuring the SONET/SDH Frame Checksum on page 514

Configuring SONET/SDH Loopback Capability on page 514

Configuring the SONET/SDH Path Trace Identifier on page 516

Configuring SONET/SDH HDLC Payload Scrambling on page 516

Configuring SONET/SDH RFC 2615 Support on page 517

Configuring SONET/SDH Defect Triggers to Be Ignored on page 518

Configuring Virtual Tributary Mapping on page 521

Configuring APS and MSP on page 522

Configuring SONET/SDH Header Byte Values

To configure values in SONET/SDH header bytes, include the bytes statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
bytes {
  c2 value;
  e1-quiet value;
  f1 value;
  f2 value;
  s1 value;
  z3 value;
  z4 value;
}
```

You can configure the following SONET/SDH header bytes:

c2—Path signal label SONET/SDH overhead byte. SONET/SDH frames use the C2 byte to indicate the contents of the payload inside the frame. SONET/SDH interfaces use the C2 byte to indicate whether the payload is scrambled. For the c2 byte, *value* can be from 0 through 255. The default value is 0xCF.

e1-quiet—Default idle byte sent on the orderwire SONET/SDH overhead bytes. The routing platform does not support the orderwire channel, and hence sends this byte continuously.

f1, f2, z3, z4—SONET/SDH overhead bytes. For these bytes, *value* can be from 0 through 255. The default value is 0x00.

s1—Synchronization message SONET/SDH overhead byte. This byte is normally controlled as a side effect of the system reference clock configuration and the state of the external clock coming from an interface if the system reference clocks have been configured to use an external reference. For the s1 byte, *value* can be from 0 through 255.

Table 41 displays JUNOS software framing bytes for several specific speeds.

Table 41: SONET/SDH Framing Bytes for Specific Speeds

| Overhead Bytes | STM4 | STM16 | STM64 | OC12 | OC48 | OC192 |
|-------------------|-------|-------|-------|-------|-------|--------|
| A1 | F6 | F6 | F6 | F6 | F6 | F6 |
| A2 | 28 | 28 | 28 | 28 | 28 | 28 |
| C1 | — | — | — | 1..12 | 1..48 | 1..192 |
| H1/H2 | 6A0A | 6A0A | 6A0A | 620A | 620A | 620A |
| Z0 | 01/CC | 01/CC | 01/CC | — | — | — |
| Concatenated mode | 93FF | 93FF | 93FF | 93FF | 93FF | 93FF |

When you configure SONET/SDH header bytes, note the following:

The C2 byte is the path signal label. If the C2 byte value on an interface does not match the C2 byte value on the remote interface, the path label mismatch (PLM-P) or unequipped (UNEQ-P) alarm might occur.

When you configure SONET/SDH OC48 interfaces for channelized (multiplexed) mode (by including the no-concatenate statement at the [edit chassis fpc slot-number pic pic-number] hierarchy level), the bytes f1 statement has no effect.

Currently, the bytes e1-quiet statement is ignored if you include it in the configuration.

The bytes f2, bytes z3, bytes z4, and path-trace options work correctly on channel 0 and work in the transmit direction only on channels 1, 2, and 3.

For DS3 channels on a channelized OC12 interface, the bytes f1, bytes f2, bytes z3, and bytes z4 options have no effect.

The bytes s1 option is supported only for channel 0; it is ignored if configured on channels 1 through 11. The bytes s1 value configured on channel 0 applies to all channels on the interface.

Embedded operations channel (EOC) D1, D2, and D3 bytes are not supported.

Configuring an Incrementing STM ID

When configured in SDH framing mode, SONET/SDH interfaces on a Juniper Networks routing platform might not interoperate with some older versions of ADMs or regenerators that require an incrementing STM ID.

Current SDH standards specify a set of $3*n$ overhead bytes in an $STMn$ that includes the J0 section trace byte. The rest are essentially unused (spare Z0) and contain hexadecimal values (0x01, 0xCC, 0xCC ... 0xCC). The older version of the standard specified that the same set of bytes should contain an incrementing sequence: 1, 2, 3, ..., $3*n$. Their use was still unspecified although they might have been used to assist in frame alignment. You can configure an incrementing STM ID to enable your Juniper Networks routing platform to interoperate with older equipment that relies on these bytes for frame alignment.

The STM identifier has a precise definition in the SDH specifications. In ITU-T Recommendation G.707, *Network node interface for the synchronous digital hierarchy (SDH)* (03/96), Section 9.2.2.2.

You can explicitly configure an incrementing STM ID rather than a static one in the SDH overhead by including the z0-increment statement at the [edit interfaces interface-name sonet-options] hierarchy level. You should include this statement only for SDH mode; do not use it for SONET mode.

```
[edit interfaces so-fpc/pic/port sonet-options]
z0-increment;
```

To explicitly disable incrementing of the STM ID, include the following statement:

```
[edit interfaces so-fpc/pic/port sonet-options]
no-z0-increment;
```

Configuring the SONET/SDH Frame Checksum

By default, SONET/SDH interfaces use a 16-bit frame checksum. You can configure a 32-bit checksum, which provides more reliable packet verification. However, some older equipment might not support 32-bit checksums.

To configure a 32-bit checksum, include the `fcs` statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
fcs 32;
```

To return to the default 16-bit frame checksum, delete the `fcs 32` statement from the configuration:

```
[edit]
user@host# delete interfaces so-fpc/pic/port sonet-options fcs 32
```

To explicitly configure a 16-bit checksum, include the `fcs` statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
fcs 16;
```

On a channelized OC12 interface, the `sonet-options fcs` statement is not supported. To configure the frame checksum sequence (FCS) on each DS3 channel, you must include the `t3-options fcs` statement in the configuration for each channel.

Configuring SONET/SDH Loopback Capability

To configure loopback capability on a SONET/SDH interface, include the `loopback` statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
loopback (local | remote);
```

To exchange BERT patterns between a local routing platform and a remote routing platform, you include the `loopback remote` statement in the interface configuration at the remote end of the link. From the local routing platform, you issue the `test interface` command.

For more information about configuring BERT, see “Configuring BERT Properties” on page 86. For more information about using operational mode commands to test interfaces, see the *JUNOS Network and Services Interfaces Command Reference*.

To turn off the loopback capability, remove the `loopback` statement from the configuration:

```
[edit]
user@host# delete interfaces so-fpc/pic/port sonet-options loopback
```

For channel 0 on channelized interfaces only, you can include the loopback statement at the [edit interfaces *interface-name* *interface-type-options*] hierarchy level. The loopback setting configured for channel 0 applies to all channels on the channelized interface. The loopback statement is ignored if you include it at this hierarchy level in the configuration of other channels. To configure loopbacks on individual channels, you must include the *channel-type-options* loopback statement in the configuration for each channel. This allows each channel to be put in loopback mode independently.

For example, for DS3 channels on a channelized OC12 interface, the sonet-options loopback statement is supported only for channel 0; it is ignored if included in the configuration for channels 1 through 11. The SONET/SDH loopback configured for channel 0 applies to all 12 channels equally. To configure loopbacks on the individual DS3 channels, you must include the t3-options loopback statement in the configuration for each channel. This allows each DS3 channel can be put in loopback mode independently.

You can determine whether there is an internal problem or an external problem by checking the error counters in the output of the show interface *interface-name* extensive command:

```
user@host> show interfaces so-fpc/pic/port extensive
```

Example: Configuring SONET/SDH Loopback Capability

To determine whether a problem is internal or external, loop packets on both the local and the remote routing platform. To do this, include the no-keepalives and encapsulation cisco-hdlc statements at the [edit interfaces *interface-name*] hierarchy level, and the loopback local statement at the [edit interfaces *interface-name* sonet-options] hierarchy level. With this configuration, the link stays up, so you can loop ping packets to a remote routing platform. The loopback local statement causes the interface to loop within the PIC just before the data reaches the transceiver.

```
[edit interfaces]
so-1/0/0 {
  no-keepalives;
  encapsulation cisco-hdlc;
  sonet-options {
    loopback local;
  }
  unit 0 {
    family inet {
      address 10.100.100.1/24;
    }
  }
}
```

Configuring the SONET/SDH Path Trace Identifier

The SONET/SDH *path trace identifier* is a text string that identifies the circuit. If the string contains spaces, enclose it in quotation marks.

By default, the JUNOS software uses the router and interface names for the path trace identifier. Depending on the router and interface names, the default path trace identifier might be longer than 16 bytes. The SDH standards define a maximum 16-byte path trace. For this reason, the default path trace identifier might be truncated in SDH mode. You can prevent the path trace identifier from being truncated in SDH mode by configuring a path trace identifier that is under 16-bytes long. In SONET mode, a path trace identifier can be up to 64-bytes long.

For DS3 channels on a channelized OC12 interface, you can configure a unique path trace for each of the 12 channels. Each path trace can be up to 16 bytes. For channels on a channelized OC12 intelligent queuing (IQ) interface, each path trace can be up to 64 bytes.

To configure a path trace identifier, include the path-trace statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
path-trace trace-string;
```

A common convention is to use the circuit identifier as the path trace identifier.

To display the local router's path trace identifier, issue the show interfaces command on the remote router.

Configuring SONET/SDH HDLC Payload Scrambling

SONET/SDH HDLC payload scrambling, which is enabled by default, provides better link stability. Both sides of a connection must either use or not use scrambling.



NOTE: HDLC payload scrambling conflicts with traffic shaping configured using leaky bucket properties. If you configure leaky bucket properties, you must disable payload scrambling, because the JUNOS software rejects configurations that have both features enabled. For more information, see “Configuring Receive and Transmit Leaky Bucket Properties” on page 533.

On a channelized OC12 interface, the sonet-options payload-scrambler statement is ignored. To configure scrambling on the DS3 channels on the interface, include the t3-options payload-scrambler statement in the configuration for each DS3 channel.

To disable HDLC payload scrambling, include the no-payload-scrambler statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
no-payload-scrambler;
```

To return to the default, that is, to re-enable payload scrambling, delete the no-payload-scrambler statement from the configuration:

```
[edit]
user@host# delete interfaces so-fpc/pic/port sonet-options
no-payload-scrambler
```

To explicitly enable payload scrambling, include the payload-scrambler statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
payload-scrambler;
```

Configuring SONET/SDH RFC 2615 Support

RFC 2615, *PPP over SONET/SDH*, requires certain C2 header byte and FCS settings that vary from the default values configured in accordance with RFC 1619 (the previous version of RFC 2615). The newer values are optimized for stronger error detection, especially when combined with payload scrambling at higher bit rate links.

Table 42 shows the older (RFC 1619) and newer (RFC 2615) values, together with the Juniper Networks default values.

Table 42: SONET/SDH Default Settings

| Value | RFC 1619 | Default | RFC 2615 |
|--------------------------|----------|---------|----------|
| SONET/SDH C2 header byte | 0XCF | 0XCF | 0X16 |
| Frame checksum (bit) | 16 | 16 | 32 |
| Payload scrambling | n/a | Enabled | Enabled |

To enable support for the RFC 2615 features, include the rfc-2615 statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options]
rfc-2615;
```

Configuring SONET/SDH Defect Triggers to Be Ignored

A trigger is a defect alarm that causes a physical interface to be marked down. By default, all defects are honored with no hold time. For SONET/SDH and ATM over SONET/SDH interfaces only, you can configure individual triggers to ignore a defect, honor a defect, and apply up and down hold timers to the defect.

Table 43 lists the defects you can configure.

Table 43: SONET/SDH and ATM Active Alarms and Defects

| Alarm | Description |
|-----------------|--------------------------------------|
| Physical | |
| pll | Phase-locked loop out of lock |
| lol | Loss of light |
| Section | |
| lof | Loss of frame |
| los | Loss of signal |
| Line | |
| ais-l | Alarm indication signal—line |
| rfl-l | Remote failure indication—line |
| ber-sd | Bit error rate defect-signal degrade |
| ber-sf | Bit error rate fault-signal fail |
| Path | |
| ais-p | Alarm indication signal—path |
| locd (ATM only) | Loss of cell delineation |
| lop-p | Loss of pointer—path |
| plm-p | Payload label mismatch |
| rfl-p | Remote failure indication—path |
| uneq-p | Path unequipped |

To configure defects to be ignored, include the trigger statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces interface-name sonet-options]
trigger {
  defect ignore;
}
```

If you configure a defect to be ignored, that defect does not contribute to the interface being marked down or up.

After you configure a defect to be ignored, the JUNOS software reevaluates the state of the defect on the interface. If the defect is outstanding and has caused the interface to be marked down, the interface is marked up.

When you configure a trigger on a low-level defect—for example, an LOS—only the low-level defect is affected. Higher-level defects that might result from the lower-level defect are not affected by the low-level trigger configuration. Therefore, you must configure higher-level defects as well.

Applying Up and Down Hold Timers to the Defect

By default, an interface is marked down as soon as a defect is detected, and is marked up as soon as the defect is absent. You might want to apply hold times to defects for the following reasons:

- To prevent route flaps from happening before a defect has been outstanding for a longer period than would be expected for an Automatic Protection Switching (APS) cutover

- To reduce the number of interface transitions

When you apply a “down” hold time to a defect, the defect must be present for at least the hold-time period before the interface is marked down. When you apply an “up” hold time to a defect, the defect must remain absent for at least the hold-time period before the interface is marked up, assuming no other defect is outstanding.

When you configure hold timers and the interface goes from up to down, the interface transition is not advertised to the rest of the system until the interface has remained down for the hold-time period. Similarly, when an interface goes from down to up, the interface transition is not advertised until the interface has remained up for the hold-time period.

To configure hold timers, include the hold-time statement at the [edit interfaces *interface-name* sonet-options trigger *defect*] hierarchy level:

```
[edit interfaces interface-name sonet-options trigger defect]
hold-time up milliseconds down milliseconds;
```

The time can be a value from 1 through 65,534 milliseconds.

When you configure defect hold times, you should note the following:

You can configure an up hold time, a down hold time, or both.

Each interface on a SONET/SDH PIC controls certain aspects of the SONET/SDH overhead. For example, when you configure an OC48 PIC to be nonconcatenated, four interfaces are created. Each interface has its own path overhead. However, all four path interfaces share the same physical, section, and line overhead. This means the following:

Each interface's path trigger configuration is honored.

The physical, section, and line trigger configuration for the primary interface (*so-fpc/pic/slot:0*) is applied to all four interfaces.

Therefore, if you configure the *so-fpc/pic/slot:0* interface to have a hold time for the LOS trigger, when an LOS event occurs, all four interfaces remain up until the trigger expires, and then all four interfaces are marked down.

The hold timers on the SONET/SDH defects are applied in addition to any other hold timers you configure on the interface. For example, if an interface is up and you configure a SONET/SDH trigger down hold time of 100 milliseconds and an interface down hold time of 250 milliseconds, when the SONET/SDH defect occurs, the SONET/SDH trigger timer starts. After 100 milliseconds, assuming the defect is still present, the SONET/SDH defect starts the 250 millisecond down timer. After this has expired and again assuming the defect is still outstanding, the interface will be marked down. For more information about interface hold timers, see "Damping Interface Transitions" on page 89.

Some defects are reported through a periodic poll (once every second). For these defects, there could be up to one second lost before the defect is detected and the hold timer is started. The hold timer expires in precisely the amount of time configured. At that point, the existence of the defect is checked again and the interface is marked up or down accordingly. These defects are as follows:

- lol
- pll
- ber-sf
- ber-sd

We recommend the following settings:

Configure SONET/SDH defect timers on no more than 64 interfaces per FPC.

Configure a combined up hold time and down hold time for a SONET/SDH defect to be at least 100 milliseconds.

Example: Configuring SONET/SDH Defects to Be Ignored

Prevent an LOS from bringing down an interface. An LOS can lead to the following defects:

```

AIS-L

LOF

PLL

RFI-L

RFI-P

[edit interfaces sonet-options trigger]
  ais-l ignore;
  lof ignore;
  los ignore;
  pll ignore;
  rfi-l ignore;
  rfi-p ignore;
}

```

Configuring Virtual Tributary Mapping

You can configure virtual tributary mapping to use KLM mode or ITU-T mode. By default, virtual tributary mapping uses KLM mode.

For the Channelized STM1 IQ PIC, you can configure virtual tributary mapping by including the `vtmapping` statement at the `[edit interfaces cau4-fpc/pic/port sonet-options]` hierarchy level:

```

[edit interfaces cau4-fpc/pic/port sonet-options]
  vtmapping (klm | itu-t);

```

For the STM1 PIC, you can configure virtual tributary mapping by including the `vtmapping` statement at the `[edit chassis fpc slot-number pic pic-number]` hierarchy level:

```

[edit chassis fpc slot-number pic pic-number]
  vtmapping (klm | itu-t);

```

Table 27 on page 298 lists the KLM mappings used by the Channelized STM1-to-E1 PIC interfaces.

Configuring APS and MSP

Automatic Protection Switching (APS) is used by SONET add/drop multiplexers (ADMs) to protect against circuit failures. The JUNOS implementation of APS allows you to protect against circuit failures between an ADM and one or more routing platforms, and between multiple interfaces in the same routing platform. When a circuit or routing platform fails, a backup immediately takes over.



NOTE: For SDH interfaces, the JUNOS software supports multiplex section protection (MSP). You configure MSP with the same CLI statements you use to configure APS.

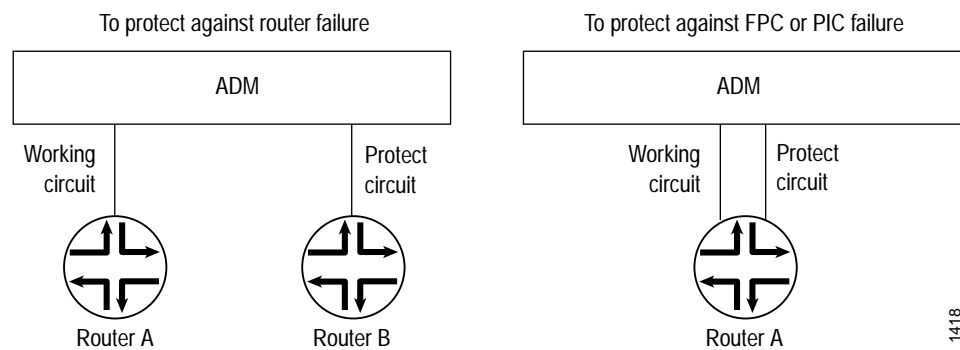
The JUNOS software supports APS 1+ 1 switching, either revertive or nonrevertive mode, and bidirectional mode only (although you can configure interoperability with line-terminating equipment [LTE] provisioned for unidirectional mode). The JUNOS software does not transmit identical data on the working and protect circuits, as the APS specification requires for 1+ 1 switching, but this causes no operational impact.

For DS3 channels on a channelized OC12 interface, you can configure APS on channel 0 only. If you configure APS on channels 1 through 11, it is ignored.

With APS and MSP, you configure two circuits, a *working circuit* and a *protect circuit*. Normally, traffic is carried on the working circuit (that is, the working circuit is the active circuit), and the protect circuit is disabled. If the working circuit fails or degrades, or if the working router fails, the ADM and the protect router switch the traffic to the protect circuit, and the protect circuit becomes the active circuit.

To configure APS or MSP, you configure a *working* and a *protect* circuit, as shown in Figure 32. To protect against a routing platform failure, you connect two routing platforms to the ADM, configuring one of them as the working router and the second as the protect router. To protect against a PIC or FPC failure, you connect one routing platform to the ADM through both the working and protect circuits, configuring one of the PICs or FPCs as the working circuit and the second as the protect circuit.

Figure 32: APS/MSP Configuration Topologies



1418

To configure APS or MSP, include the `aps` statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces interface-name sonet-options]
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;
  switching-mode (bidirectional | unidirectional);
  working-circuit group-name;
}
```

You can configure the following APS properties:

Configuring Basic APS Support on page 524

Configuring Switching Between the Working and Protect Circuits on page 525

Configuring Revertive Mode on page 526

Configuring Unidirectional Switching Mode Support on page 526

Configuring APS Timers on page 527

Configuring APS Load Sharing Between Circuit Pairs on page 528

Example: Configuring APS Load Sharing Between Circuit Pairs on page 529



NOTE: This implementation of APS is not supported on Layer 2 circuits. For Layer 2 circuits, you configure APS by including the `protect-interface` statement. You can include this statement at the following hierarchy levels:

```
[edit logical-routers logical-router-name protocols l2circuit neighbor neighbor-id
interface interface-name]
```

```
[edit protocols l2circuit neighbor neighbor-id interface interface-name]
```

For more information and a configuration example, see the *JUNOS VPNs Configuration Guide*.

Configuring Basic APS Support

To set up a basic APS configuration, configure one interface to be the working circuit and a second to be the protect circuit. If you are using APS to protect against routing platform failure, configure one interface on each routing platform. If you are using APS to protect against FPC failure, configure two interfaces on the routing platform, one on each FPC.

For each working–protect circuit pair, configure the following:

Group name—Creates the association between the two circuits. Configure the same group name for both the working and protect routers.

Authentication key—You configure this on both interfaces. Configure the same key for both the working and protect routers.

Address of the other interface on the other routing platform—If you are configuring one routing platform to be the working router and a second to be the protect router, you must configure the address of the remote interface. You configure this on one or both of the interfaces.

The address you specify for the neighbor must never be routed through the interface on which APS is configured, or instability will result. APS neighbor only applies to inter-routing platform configurations. We strongly recommend that you directly connect the working and protect routers and that you configure the interface address of this shared network as the neighbor address.

The working and protect configurations on the routing platforms must match the circuit configurations on the ADM; that is, the working router must be connected to the ADM's working circuit and the protect router must be connected to the protect circuit.

To set up a basic APS configuration, include the following statements at the [edit interfaces *interface-name* sonet-options] hierarchy level:

On the Working Circuit [edit interfaces *so-fpc/pic/port* sonet-options]
 aps {
 working-circuit *group-name*;
 authentication-key *key*;
 neighbor *address*; # Include if protect circuit is on a different routing platform
 }

On the Protect Circuit aps {
 protect-circuit *group-name*;
 authentication-key *key*;
 neighbor *address*; # Include if working circuit is on a different routing platform
 }

Example: Configuring Basic APS Support

Configure Router A to be the working router and Router B to be the protect router.

| | |
|--|--|
| On Router A (the Working Router) | <pre>[edit interfaces so-6/1/1 sonet-options] aps { working-circuit San-Jose; authentication-key "\$9\$B2612345"; }</pre> |
| On Router B (the Protect Circuit) | <pre>[edit interfaces so-0/0/0 sonet-options] aps { protect-circuit San-Jose; authentication-key "\$9\$B2612345"; neighbor 192.168.1.2; # Address of Router A on the link between A and B }</pre> |
| On a Single Platform, One Interface as the Working Circuit and Another Interface as the Protect Circuit | <pre>[edit interfaces so-2/1/1 sonet-options] aps { working-circuit Hayward; authentication-key blarney; } [edit interfaces so-3/0/2 sonet-options] aps { protect-circuit Hayward; authentication-key blarney; }</pre> |

Configuring Switching Between the Working and Protect Circuits

When there are multiple reasons to switch between the working and protect circuits, a priority scheme is used to decide which circuit to use. The routing platforms and the ADM might automatically switch traffic between the working and protect circuits because of circuit and routing platform failures. You can also choose to switch traffic manually between the working and protect circuits. There are three priority levels of manual configuration, listed here in order from lowest to highest priority:

Request (also known as manual switch)—Overridden by signal failures, signal degradations, or any higher-priority reasons.

Force (also known as forced switch)—Overrides manual switches, signal failures, and signal degradation.

Lockout (also known as lockout of protection)—Do not switch between the working and protect circuits.

A routing platform failure is considered to be equivalent to a signal failure on a circuit.

To perform a manual switch, include the request statement at the [edit interfaces *interface-name* sonet-options aps] hierarchy level. This statement is honored only if there are no higher-priority reasons to switch.

```
[edit interfaces so-fpc/pic/port sonet-options aps]
request (protect | working);
```

When the working circuit is operating in nonrevertive mode, use the `request working` statement to switch the circuit manually to being the working circuit or to override the revert timer.

To perform a forced switch, include the `force` statement at the `[edit interfaces interface-name sonet-options aps]` hierarchy level. This statement is honored only if there are no higher-priority reasons to switch. This configuration can be overridden by a signal failure on the protect circuit, thus causing a switch to the working circuit.

```
[edit interfaces so-fpc/pic/port sonet-options aps]
force (protect | working);
```

To configure a lockout of protection, forcing the use of the working circuit and locking out the protect circuit regardless of anything else, include the `lockout` statement at the `[edit interfaces interface-name sonet-options aps]` hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options aps]
lockout;
```

Configuring Revertive Mode

By default, APS is nonrevertive, which means that if the protect circuit becomes active, traffic is not switched back to the working circuit unless the protect circuit fails or you manually configure a switch to the working circuit. In revertive mode, traffic is automatically switched back to the working circuit.

You should configure the ADM and routing platforms consistently with regard to revertive or nonrevertive mode.

To configure revertive mode, include the `revert-time` statement, specifying the amount of time to wait after the working circuit has again become functional before making the working circuit active again:

```
[edit interfaces so-fpc/pic/port sonet-options aps]
revert-time seconds;
```

If you are using nonrevertive APS, you can use the `request working` statement to switch the circuit manually to being the working circuit or to override the revert timer (configured with the `revert-time` statement).

Configuring Unidirectional Switching Mode Support

For unchannelized OC3, OC12, and OC48 SONET/SDH interfaces on T-series platforms only, you can configure interoperation with SONET/SDH LTE that is provisioned for unidirectional linear APS in 1+ 1 architecture.

By default, APS supports only SONET/SDH LTE that is provisioned for bidirectional mode.

In bidirectional switching mode, the working interface switches to the protect interface for both receipt and transmission of data, regardless of whether the signal failure is in the transmit or receive direction.

In true unidirectional mode, the working interface switches to the protect interface only for the direction in which signal failure occurs; for example, if there is a signal failure in the transmit direction, the working interface switches over to the protect interface for transmission but not receipt of data. When the protect interface operates in unidirectional mode, the working and protect interfaces must cooperate to operate the transmit and receive interfaces in a bidirectional fashion.

The JUNOS software does not support true unidirectional mode. Instead the software supports interoperability with SONET/SDH LTE provisioned for unidirectional switching. This means that the SONET/SDH LTE on the router receives and transmits on one interface, even when you configure unidirectional support. The JUNOS implementation of unidirectional mode support allows the router to do the following:

- Accept a unidirectional mode as valid

- Trigger the peer (ADM) selector to switch receive from working interface to protect interface or the reverse

- Not send reverse requests to the far end (ADM)

To configure unidirectional mode support, include the `switching-mode unidirectional` statement, at the `[edit interfaces interface-name sonet-options aps]` hierarchy level:

```
[edit interfaces interface-name sonet-options aps]
switching-mode unidirectional;
```



NOTE: On interfaces with unidirectional APS support configured, revertive mode and load sharing between circuits are not supported.

To restore the default behavior, include the `switching-mode bidirectional` statement, at the `[edit interfaces interface-name sonet-options aps]` hierarchy level:

```
[edit interfaces interface-name sonet-options aps]
switching-mode bidirectional;
```

Configuring APS Timers

The protect and working routers periodically send packets to their neighbors to advertise that they are operational. By default, these advertisement packets are sent every 1000 milliseconds. A routing platform considers its neighbor to be operational for a period, called the hold time, that is, by default, three times the advertisement interval. If the protect router does not receive an advertisement packet from the working router within the hold time configured on the protect router, the protect router assumes that the working router has failed and becomes active.

APS is symmetric; either side of a circuit can time out the other side (for example, when detecting a crash of the other). Under normal circumstances, the failure of the protect router does not cause any changes because the traffic is already moving on the working router. However, if you had configured request protect and the protect router failed, the working router would enable its interface.

To modify the advertisement interval, include the `advertise-interval` at the [edit interfaces *interface-name* sonet-options aps] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options aps]
advertise-interval milliseconds;
```

To modify the hold time, include the `hold-time` at the [edit interfaces *interface-name* sonet-options aps] hierarchy level:

```
[edit interfaces so-fpc/pic/port sonet-options aps]
hold-time milliseconds;
```

The advertisement intervals and hold times on the protect and working routers can be different.

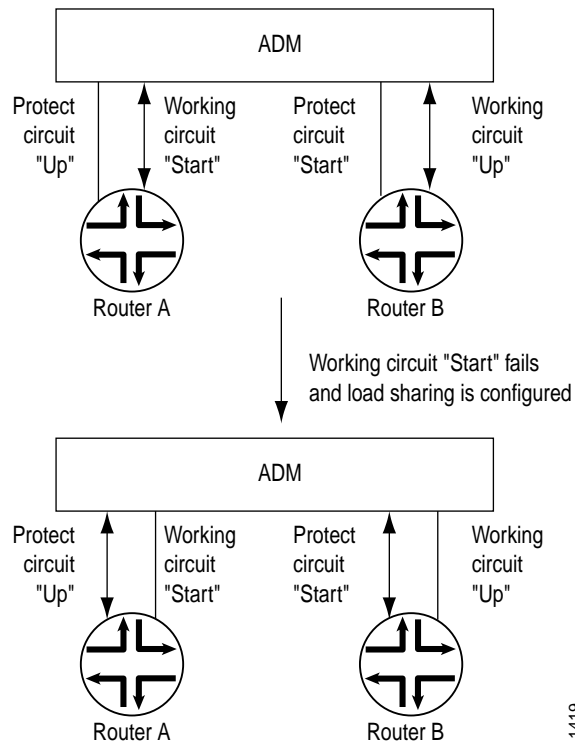
Configuring APS Load Sharing Between Circuit Pairs

When two routing platforms are connected to a single ADM, you can have them back up each other on two different pairs of circuits. This arrangement provides load balancing between the routing platforms if one of the working circuits fails.

Figure 33 illustrates load sharing between circuits on two routing platforms. Router A has a working circuit “Start” and a protect circuit “Up,” and Router B has a working circuit “Up” and a protect circuit “Start.” Under normal circumstances, Router A carries the “Start” circuit traffic and Router B carries the “Up” circuit traffic. If the working circuit “Start” were to fail, Router B would end up carrying all the traffic for both the “Start” and “Up” circuits.

To balance the load between the circuits, you pair the two circuits. In this case, you pair the “Start” and “Up” circuits. Then, if the working circuit “Start” fails, the two routing platforms automatically switch the “Up” traffic from the working to the protect circuit so that each routing platform is still carrying only one circuit’s worth of traffic. That is, the working circuit on Router A would be “Up” and the working circuit on Router B would be “Start.”

Figure 33: APS Load Sharing Between Circuit Pairs



To configure load sharing between two working–protect circuit pairs, include the paired-group statement when configuring one of the circuits on one of the routing platforms. In this statement, the *group-name* is the name of the group you assigned to one of the circuits with the working-circuit and protect-circuit statements. The JUNOS software automatically configures the remainder of the load-sharing setup based on the group name.

```
[edit interfaces so-fpc/pic/port sonet-options aps]
paired-group group-name;
```

Example: Configuring APS Load Sharing Between Circuit Pairs

Configure APS load sharing to match the configuration shown in Figure 33 on page 529:

```
On Router A [edit interfaces so-7/0/0 sonet-options aps]
user@host# set working-circuit start
[edit interfaces so-7/0/0 sonet-options aps]
user@host# set authentication-key linsey
[edit interfaces so-7/0/0 sonet-options aps]
user@host# set paired-group "Router A-Router B"
...
[edit interfaces so-0/0/0 sonet-options aps]
user@host# set protect-circuit up
[edit interfaces so-0/0/0 sonet-options aps]
user@host# set authentication-key woolsey
[edit interfaces so-0/0/0 sonet-options aps]
user@host# set paired-group "Router A-Router B"
```

```

On Router B [edit interfaces so-1/0/0 sonet-options aps]
user@host# set working-circuit up
[edit interfaces so-1/0/0 sonet-options aps]
user@host# set authentication-key woolsey
[edit interfaces so-1/0/0 sonet-options aps]
user@host# set paired-group "Router A-Router B"
...
[edit interfaces so-6/0/0 sonet-options aps]
user@host# set protect-circuit start
[edit interfaces so-6/0/0 sonet-options aps]
user@host# set authentication-key linsey
[edit interfaces so-6/0/0 sonet-options aps]
user@host# set paired-group "Router A-Router B"

```

Configuring the Media MTU

The default media MTU size used on a physical interface depends on the encapsulation being used on that interface. For a listing of MTU sizes for each encapsulation type, see “Configuring the Media MTU” on page 67. For information about configuring the encapsulation on an interface, see “Configuring Interface Encapsulation” on page 535.

To modify the default media MTU size for a physical interface, include the `mtu` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
mtu bytes;
```

If you change the size of the media MTU, you must ensure that the size is equal to or greater than the sum of the protocol MTU and the encapsulation overhead. You configure the protocol MTU by including the `mtu` 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]
```

For more information, see “Setting the Protocol MTU” on page 117.

Enabling Passive Monitoring on SONET/SDH Interfaces

The Monitoring Services I and Monitoring Services II PICs are designed to enable IP services. If you have a Monitoring Services PIC and a SONET/SDH PIC installed in an M160, M40e, or T-series routing platform, you can monitor IPv4 traffic from another routing platform.

On SONET/SDH interfaces, you enable packet flow monitoring by including the `passive-monitor-mode` statement at the `[edit interfaces so-fpc/pic/port unit logical-unit-number]` hierarchy level:

```
[edit interfaces so-fpc/pic/port unit logical-unit-number]
passive-monitor-mode;
```

If you include the `passive-monitor-mode` statement in the configuration, the SONET/SDH interface does not send keepalives or alarms, and does not participate actively on the network.

On monitoring services interfaces, you enable packet flow monitoring by including the `family` statement at the `[edit interfaces mo-fpc/pic/port unit logical-unit-number]` hierarchy level, specifying the `inet` option:

```
[edit interfaces mo-fpc/pic/port unit logical-unit-number]
family inet;
```

For conformity with `cflowd` record structure, you must include the `receive-options-packets` and `receive-ttl-exceeded` statements at the `[edit interfaces mo-fpc/pic/port unit logical-unit-number family inet]` hierarchy level:

```
[edit interfaces mo-fpc/pic/port unit logical-unit-number family inet]
receive-options-packets;
receive-ttl-exceeded;
```

For the monitoring services interface, you can configure multiservice physical interface properties. For more information, see “Configuring Multiservice Physical Interface Properties” on page 89 and the *JUNOS Services Interfaces Configuration Guide*.

Removing MPLS Labels from Incoming Packets

The JUNOS software can forward only IPv4 packets to a Monitoring Services PIC. IPv4 packets with MPLS labels cannot be forwarded to a Monitoring Services PIC. By default, if packets with MPLS labels are forwarded to the Monitoring Services PIC, they are discarded. To monitor packets with MPLS labels, you must remove the MPLS labels as the packets arrive on the interface.

You can remove up to two MPLS labels from an incoming packet by including the `pop-all-labels` statement at the `[edit interfaces interface-name sonet-options mpls]` hierarchy level:

```
[edit interfaces interface-name sonet-options mpls]
pop-all-labels {
    required-depth number;
}
```

By default, the `pop-all-labels` statement takes effect for incoming packets with one or two labels. You can specify the number of MPLS labels an incoming packet must have for the `pop-all-labels` statement to take effect by including the `required-depth` statement at the `[edit interfaces interface-name atm-options mpls pop-all-labels]` hierarchy level:

```
[edit interfaces interface-name atm-options mpls pop-all-labels]
required-depth number;
```

The required depth can be 1, 2, or [1 2]. If you include the required-depth 1 statement, the pop-all-labels statement takes effect for incoming packets with one label only. If you include the required-depth 2 statement, the pop-all-labels statement takes effect for incoming packets with two labels only. If you include the required-depth [1 2] statement, the pop-all-labels statement takes effect for incoming packets with one or two labels. A required depth of [1 2] is equivalent to the default behavior of the pop-all-labels statement.

When you remove MPLS labels from incoming packets, note the following:

The pop-all-labels statement has no effect on IP packets with three or more MPLS labels.

When you enable MPLS label removal, you must configure all ports on a PIC with the same label popping mode and required depth.

You use the pop-all-labels statement to enable passive monitoring applications, not active monitoring.

You cannot apply MPLS filters or accounting to the MPLS labels because the labels are removed as soon as the packet arrives on the interface.

Configuring the Clock Source

For interfaces such as SONET/SDH that can use different clock sources, you can configure the source of the transmit clock on each interface. The source can be internal (also called line timing or normal) or external (also called loop timing). The default source is internal, which means that each interface uses the routing platform's internal stratum 3 clock.

For DS3 channels on a channelized OC12 interface, the clocking statement is supported only for channel 0; it is ignored if included in the configuration of channels 1 through 11. The clock source configured for channel 0 applies to all channels on the channelized OC12 interface. The individual DS3 channels use a gapped 45-MHz clock as the transmit clock. For more information, see "Clock Sources on Channelized Interfaces" on page 250.



NOTE: On channelized STM1 interfaces, you should configure the clock source at one side of the connection to be internal (the default JUNOS configuration) and configure the other side of the connection to be external.

To configure loop timing on an interface, include the clocking external statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]  
clocking external;
```

To explicitly configure line timing on an interface, include the clocking internal statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]  
clocking internal;
```

Configuring Receive and Transmit Leaky Bucket Properties

Congestion control is particularly difficult in high-speed networks with high volumes of traffic. When congestion occurs in such a network, it is usually too late to react. You can avoid congestion by regulating the flow of packets into your network. Smoother flows prevent bursts of packets from arriving at (or being transmitted from) the same interface and causing congestion.

For all interface types except ATM, Fast Ethernet, Gigabit Ethernet, and channelized IQ, you can configure leaky bucket properties, which allow you to limit the amount of traffic received on and transmitted by a particular interface. You effectively specify what percentage of the interface's total capacity can be used to receive or transmit packets. You might want to set leaky bucket properties to limit the traffic flow from a link that is known to transmit high volumes of traffic.



NOTE: Instead of configuring leaky bucket properties, you can limit traffic flow by configuring policers. Policers work on all interfaces. For more information, see the *JUNOS Policy Framework Configuration Guide*.

The leaky bucket is used at the host-network interface to allow packets into the network at a constant rate. Packets might be generated in a bursty manner, but after they pass through the leaky bucket, they enter the network evenly spaced. In some cases, you might want to allow short bursts of packets to enter the network without smoothing them out. By controlling the number of packets that can accumulate in the bucket, the threshold property controls burstiness. The maximum number of packets entering the network in t time units is $\text{threshold} + \text{rate} * t$.

By default, leaky buckets are disabled and the interface can receive and transmit packets at the maximum line rate.

For each DS3 channel on a channelized OC12 interface, you can configure unique receive and transmit buckets.



NOTE: HDLC payload scrambling conflicts with traffic shaping configured using leaky bucket properties. If you configure leaky bucket properties, you must disable payload scrambling, because the JUNOS software rejects configurations that have both features enabled. For more information, see "Configuring SONET/SDH HDLC Payload Scrambling" on page 516.

To configure leaky bucket properties, include one or both of the `receive-bucket` and `transmit-bucket` statements at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
receive-bucket {
  overflow (discard | tag);
  rate percentage;
  threshold bytes;
}
transmit-buckets {
  overflow discard;
  rate percentage;
  threshold bytes;
}
```

In the `rate` statement, specify the percentage of the interface line rate that is available to receive or transmit packets. The percentage can be a value from 0 (none of the interface line rate is available) to 100 (the maximum interface line rate is available). For example, when you set the line rate to 33, the interface receives or transmits at one third of the maximum line rate.

In the `threshold` statement, specify the bucket threshold, which controls the burstiness of the leaky bucket mechanism. The larger the value, the more bursty the traffic, which means that over a very short amount of time the interface can receive or transmit close to line rate, but the average over a longer time is at the configured bucket rate. The threshold can be a value from 0 through 16,777,215 bytes. For ease of entry, you can enter *number* either as a complete decimal number or as a decimal number followed by the abbreviation k (1,000) or m (1,000,000). For example, the entry `threshold 2m` corresponds to a threshold of 2,000,000 bytes.

In the `overflow` option, specify how to handle packets that exceed the threshold:

`tag`—(receive-bucket only) Tag, count, and process received packets that exceed the threshold.

`discard`—Discard received packets that exceed the threshold. No counting is done.

Damping Interface Transitions

By default, when an interface changes from being up to being down, or from down to up, this transition is advertised immediately to the hardware and the JUNOS software. In some situations—for example, when an interface is connected to an add-drop multiplexer (ADM) or wavelength-division multiplexer (WDM), or to protect against SONET/SDH framer holes—you might want to damp interface transitions. This means not advertising the interface's transition until a certain period of time has transpired.

When you have damped interface transitions and the interface goes from up to down, the interface is not advertised to the rest of the system as being down until it has remained down for the hold-time period. Similarly when an interface goes from down to up, it is not advertised as being up until it has remained up for the hold-time period.

To damp interface transitions, include the hold-time statement at the [edit interfaces *interface-name*] hierarchy level:

```
hold-time up milliseconds down milliseconds;
```

The time can be a value from 0 through 65,534 milliseconds. The default value is 0, which means that interface transitions are not damped. The JUNOS software advertises the transition within 100 milliseconds of the time value you specify.

Configuring Interface Encapsulation

Point-to-Point Protocol (PPP) encapsulation is the default encapsulation type for physical interfaces. You need not configure encapsulation for any physical interfaces that support PPP encapsulation. If you do not configure encapsulation, PPP is used by default. For physical interfaces that do not support PPP encapsulation, you must configure an encapsulation to use for packets transmitted on the interface. You can optionally configure an encapsulation on a logical interface, which is the encapsulation used within certain packet types.

Configuring the Encapsulation on a Physical Interface

For SONET/SDH interfaces, the physical interface encapsulation can be one of the following:

Point-to-Point Protocol (PPP)—PPP encapsulation is defined in RFC 1661, *The Point-to-Point Protocol (PPP) for the Transmission of Multiprotocol Datagrams over Point-to-Point Links*. PPP is the default encapsulation type for physical interfaces. Two related versions are supported:

Circuit cross-connect (CCC) version (ppp-ccc)—The logical interfaces do not require an encapsulation statement. When you use this encapsulation type, you can configure the ccc family only.

Translational cross-connect (TCC) version (ppp-tcc)—Similar to CCC and has the same configuration restrictions, but used for circuits with different media on either side of the connection.

Cisco HDLC—E1, E3, SONET/SDH, T1, and T3 interfaces can use Cisco HDLC encapsulation. Two related versions are supported:

CCC version (cisco-hdlc-ccc)—The logical interfaces do not require an encapsulation statement. When you use this encapsulation type, you can configure the ccc family only.

TCC version (cisco-hdlc-tcc)—Similar to CCC and has the same configuration restrictions, but used for circuits with different media on either side of the connection.

Frame Relay—Defined in RFC 1490, *Multiprotocol Interconnection over Frame Relay*. E1, E3, SONET/SDH, T1, and T3 interfaces can use Frame Relay encapsulation. Two related versions are supported:

CCC version (frame-relay-ccc)—The same as standard Frame Relay for DLCIs 0 through 511. DLCIs 512 through 1022 are dedicated to CCC. This numbering restriction does not apply to IQ interfaces. The logical interface must also have frame-relay-ccc encapsulation.

TCC version (frame-relay-tcc)—Similar to Frame Relay CCC and has the same configuration restrictions, but used for circuits with different media on either side of the connection.

To configure the encapsulation on a physical interface, include the encapsulation statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]
encapsulation (cisco-hdlc | cisco-hdlc-ccc | cisco-hdlc-tcc | frame-relay |
frame-relay-ccc | frame-relay-tcc | frame-relay-tcc | ppp | ppp-ccc | ppp-tcc);
```

When you configure a point-to-point encapsulation (such as PPP or Cisco HDLC) on a physical interface, the physical interface can have only one logical interface (that is, only one unit statement) associated with it. When you configure a multipoint encapsulation (such as Frame Relay), the physical interface can have multiple logical units, and the units can be either point to point or multipoint. Use PPP if you are running Cisco IOS Release 12.0 or later. If you need to run Cisco HDLC, the JUNOS software automatically configures an ISO family MTU of 4469 in the routing platform. This is due to an extra byte of padding used by Cisco.

For more information about physical interface encapsulation, see “Configuring the Encapsulation on a Physical Interface” on page 73.

Example: Configuring the Encapsulation on a Physical Interface

Configure PPP encapsulation on a SONET/SDH interface. The second two family statements allow IS-IS and MPLS to run on the interface.

```
[edit interfaces]
so-7/0/0 {
  encapsulation ppp;
  unit 0 {
    point-to-point;
    family inet {
      address 192.168.1.113/32 {
        destination 192.168.1.114;
      }
    }
    family iso;
    family mpls;
  }
}
```

Configuring the Encapsulation on a Logical Interface

Generally, you configure an interface’s encapsulation at the [edit interfaces *interface-name*] hierarchy level. However, for Frame Relay encapsulation, you can also configure the encapsulation type that is used inside the Frame Relay packet itself. To do this, include the encapsulation statement, specifying the `frame-relay-ccc` or `frame-relay-tcc` option:

```
encapsulation (frame-relay-ccc | frame-relay-tcc);
```

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]
```

The ATM encapsulations are defined in RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*.

With the `atm-nlpid`, `atm-cisco-nlpid`, and `atm-vc-mux` encapsulations, you can configure the `inet` family only. With the circuit cross-connect (CCC) encapsulations, you cannot configure a family on the logical interface. A logical interface cannot have `frame-relay-ccc` encapsulation unless the physical device also has `frame-relay-ccc` encapsulation. A logical interface cannot have `frame-relay-tcc` encapsulation unless the physical device also has `frame-relay-tcc` encapsulation. In addition, you must assign this logical interface a DLCI from 512 through 1022. This numbering restriction does not apply to IQ interfaces. You must configure the logical interface as point-to-point.

For more information about logical interface encapsulation, see “Configuring the Encapsulation on a Logical Interface” on page 105.

Example: Configuring SONET/SDH Interfaces

SONET/SDH interfaces can use either PPP or Cisco HDLC encapsulation. Use PPP if you are running Cisco IOS Release 12.0 or later. If you need to run Cisco HDLC, the JUNOS software automatically configures an ISO family MTU of 4469 in the routing platform. This is due to an extra byte of padding used by Cisco. The following configuration, which uses PPP encapsulation, is sufficient to get a SONET/SDH OC3 or OC12 interface up and running:

```
[edit interfaces]
so-fpc/pic/port {
  encapsulation ppp;
  unit 0 {
    family inet {
      address local-address {
        destination remote-address;
      }
    }
  }
}
```

Configuring Aggregated SONET/SDH Interfaces

The JUNOS software enables link aggregation of SONET/SDH interfaces; this is similar to Ethernet link aggregation, but is not defined in a public standard. The JUNOS software balances traffic across the member links within an aggregated SONET/SDH bundle based on the Layer 3 information carried in the packet. This implementation uses the same load balancing algorithm used for per-packet load balancing. For information about per-packet load balancing, see the *JUNOS Routing Protocols Configuration Guide*.

You configure an aggregated SONET/SDH virtual link by specifying the link number as a physical device and then associating a set of physical interfaces that have the same speed.



NOTE: The JUNOS software does not provide load balancing for multicast traffic on aggregated interfaces. If a link carrying multicast data goes down, another link carries the traffic. This provides redundancy, not more bandwidth.

By default, no aggregated SONET/SDH interfaces are created. You must define the number of aggregated SONET/SDH interfaces by including the device-count statement at the [edit chassis aggregated-devices sonet] hierarchy level:

```
[edit chassis aggregated-devices sonet]
  device-count number;
```

The maximum number of aggregated interfaces is 16. The aggregated SONET/SDH interfaces are numbered from as0 through as15. For more information, see the *JUNOS System Basics Configuration Guide*.



NOTE: SONET/SDH aggregation is proprietary to the JUNOS software and might not work with other software.

To configure aggregated SONET/SDH interfaces, assign a number for the aggregated SONET/SDH interface asx at the [edit interfaces] hierarchy level:

```
[edit interfaces]
  asx {
  ...
  }
```

The following example shows an aggregated SONET/SDH configuration:

```
[edit interfaces]
  as0 {
    aggregated-sonet-options {
      minimum-links 1;
      link-speed oc3;
    }
    unit 0 {
      family inet {
        address 10.2.11.1/32 {
          destination 10.2.11.3;
        }
      }
    }
  }
```

You also need to specify the constituent physical interfaces by including the aggregate statement at the [edit interfaces *interface-name* sonet-options] hierarchy level; for more information, see “Configuring SONET/SDH Link Aggregation” on page 540. You can optionally specify other physical properties that apply specifically to the aggregated SONET/SDH interfaces; for details, see “Configuring SONET/SDH Physical Interface Properties” on page 510. For a sample configuration, see “Example: Configuring Aggregated SONET/SDH Interfaces” on page 543.

To remove the configuration statements related to asx and set the aggregated SONET/SDH interface to down state, delete the interface from the configuration:

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

However, the aggregated SONET/SDH interface is not deleted until you delete the chassis aggregated-devices sonet device-count configuration statement.

You can configure the following aggregated SONET/SDH properties:

Configuring SONET/SDH Link Aggregation on page 540

Configuring Aggregated SONET/SDH Link Speed on page 540

Configuring Aggregated SONET/SDH Minimum Links on page 541

Configuring Filters or Sampling on Aggregated SONET/SDH Links on page 541

Configuring SONET/SDH Link Aggregation

On SONET/SDH interfaces, you can associate a physical interface with an aggregated SONET/SDH interface. To associate the interface with an aggregated SONET/SDH link, include the aggregate statement at the [edit interfaces *interface-name* sonet-options] hierarchy level:

```
[edit interfaces interface-name sonet-options]
aggregate asx;
```

x is the interface instance number and can be from 0 through 15, for a total of 16 aggregated interfaces. You should not mix SONET and SDH mode on the same aggregated interface. You must also include a statement configuring *asx* at the [edit interfaces] hierarchy level. For a sample configuration, see “Example: Configuring Aggregated SONET/SDH Interfaces” on page 543.

You can combine like interfaces only, so each physical interface in the aggregate has to be the same speed.

Configuring Aggregated SONET/SDH Link Speed

On aggregated SONET/SDH interfaces, you can set the required link speed for all interfaces included in the bundle. All interfaces that make up a bundle must be the same speed. If you include in the aggregated SONET/SDH interface an individual link that has a speed different from the speed you specify in the link-speed parameter, an error message will be logged.



NOTE: For nonconcatenated interfaces on aggregated SONET/SDH interfaces, you can configure the link speed of the aggregate to match the speed of the nonconcatenated interface. For example, an OC12 PIC can have nonconcatenated interfaces with a link speed of OC3.

To set the required link speed, include the link-speed statement at the [edit interfaces *interface-name* aggregated-sonet-options] hierarchy level:

```
[edit interfaces interface-name aggregated-sonet-options]
link-speed speed;
```

The link speed can be one of the following values:

oc3—Links are OC3c or STM1c.

oc12—Links are OC12c or STM4c.

oc48—Links are OC48c or STM16c.

oc192—Links are OC192c or STM64c.

Configuring Aggregated SONET/SDH Minimum Links

On aggregated SONET/SDH interfaces, you can set the minimum number of links that must be up for the bundle as a whole to be labeled up. By default, only one link must be up for the bundle to be labeled up.

To set the minimum number, include the `minimum-links` statement at the [edit interfaces *interface-name* aggregated-sonet-options] hierarchy level:

```
[edit interfaces interface-name aggregated-sonet-options]
minimum-links number;
```

The number can be from one through eight. The maximum number of links supported in an aggregate is eight. When 8 is specified, all configured links of a bundle must be up.

Configuring Filters or Sampling on Aggregated SONET/SDH Links

To set up firewall filters or sampling on aggregated SONET/SDH interfaces, you must configure the `asx` interface with these properties. The filters function in the same manner as on other interfaces.

To configure a filter, include the filter statement:

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

You can include this statement at the following hierarchy levels:

```
[edit interfaces asx unit logical-unit-number]
```

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

You must also configure separate statements that define the properties of the filter. For more information, see the *JUNOS Policy Framework Configuration Guide* and “Examples: Configuring Filters or Sampling on Aggregated SONET/SDH Links” on page 541.

Examples: Configuring Filters or Sampling on Aggregated SONET/SDH Links

Configure filtering on aggregated SONET/SDH interfaces:

```
[edit interfaces]
asx {
  unit 0 {
    family inet {
      address 10.2.11.1/32 {
        destination 10.2.11.3;
      }
    }
  }
}
```

```

        filter {
            input input-filter-name;
            output output-filter-name;
        }
    }
}

```

Defining the Filter

```

[edit firewall]
filter input-filter-name {
    term match-any-input {
        then {
            accept;
        }
    }
}
filter output-filter-name {
    term match-any-output {
        then {
            accept;
        }
    }
}

```

**Configuring Sampling
on an Aggregated
SONET/SDH Interface**

```

[edit interfaces]
asx {
    unit 0 {
        family inet {
            address 10.2.11.1/32 {
                destination 10.2.11.3;
            }
            filter {
                input input-sampler-name;
            }
        }
    }
}

```

**Defining the Sampling
Filter and the
Forwarding Action**

```

[edit firewall]
filter input-sampler-name {
    term match-any-input {
        then {
            sample;
            accept;
        }
    }
}

[edit forwarding-options]
sampling {
    input {
        family inet {
            rate 10000;
            run-length 1;
        }
    }
}

```

Example: Configuring Aggregated SONET/SDH Interfaces

The following configuration is sufficient to get an aggregated SONET/SDH interface up and running:

```
[edit interfaces]
as0 {
  aggregated-sonet-options {
    minimum-links 1;
    link-speed oc3;
  }
  unit 0 {
    family inet {
      address 10.2.11.1/32 {
        destination 10.2.11.3;
      }
    }
  }
}

[edit chassis]
aggregated-devices {
  sonet {
    device-count 15;
  }
}

[edit interfaces]
so-1/3/0 {
  sonet-options {
    aggregate as0;
  }
}
```

