Serial Interfaces

You can create the serial interface on T1 or E1, T3 or E3, SDH, or SONET interface. Each serial interface configuration differs based on the interface mode.

The channel identifier configuration differs based on the interface mode. For more information, refer serial interface supported modes.

Serial Interface Supported Modes

The serial interface name is specified as **interface serial***0/bay/port*. The zero specifies the slot number, bay specifies the bay number in the slot, and port specifies the port number in the bay.

The channel identifier varies depending on port type and supported port modes.

The following table details the values for the channel ID depending on the port modes:

Table 1: Channel Identifier Supported on T1 or E1 Interface

Mode	Interface	Serial Interface with supported Channel Identifier
T1 or E1	T1 or E1	Serial0/bay/port.1 The port value ranges from 0 to 11.

Table 2: Channel Identifier Supported on T3 or E3 Interface

Mode	Interface	Serial Interface with supported Channel Identifier
T3 or E3	T3 or E3	Serial0/bay/port.1 The port value ranges from 12 to 15.
CT3 or CE3	Channelized T3 or E3	Serial0/bay/port.<t1 number> Serial0/bay/port.<e1 number> T1 or E1 number specifies the VTG number with TUG number and T1 channels. The T1 or E1 number that is supported are as follows: • VTG 1/TUG2 1: T1 {1,8,15,22} • VTG 2/TUG2 2: T1 {2,9,16,23} • VTG 3/TUG2 3: T1 {3,10,17,24} • VTG 4/TUG2 4: T1 {4,11,18,25} • VTG 5/TUG2 5: T1 {5,12,19,26} • VTG 6/TUG2 6: T1 {6,13,20,27} • VTG 7/TUG2 7: T1 {7,14,21,28

Table 3: Channel Identifier Supported on SDH or SONET Interface

Mode	Interface Mode	Serial Interface with supported Channel Identifier
SONET or SDH	STS-3c or VC-4	Serial0/bay/port.<channel-id> For SONET, the <channel-id> is calculated based on the following formula: Channel-id = (start_sts_number − 1) x 28 + 1 For SDH, the <channel-id> is calculated based on the following formula: Channel-id = (start_aug4 − 1) x 28 x 3 + 1
SONET or SDH	T3 or E3	Serial0/bay/port.<channel-id> For SONET, the <channel-id> is calculated based on the following formula: Channel-id = (start_sts_number − 1) x 28 + 1 For SDH AU-4 mapping in TUG3 mode, the <channel-id> is calculated based on the following formula: Channel-id = (AUG 4 − 1) x 28 x 3 + (TUG 3 − 1) x 28 + (e1 − 1) x 7 + TUG 2 For SDH AU-3 mapping, the <channel-id> is calculated based on the following formula: Channel-id = (AUG 3 − 1) x 28 + (e1 − 1) x 7 + TUG 2

Mode	Interface Mode	Serial Interface with supported Channel Identifier
SONET or SDH	Concatenated Mode	For SONET, the <channel-id> is calculated based on the following formula: Channel-id = (start_sts_number − 1) x 28 + 1 For SDH, the <channel-id> is calculated based on the following formula: Channel-id = (start_aug4 −1) x 28 x 3 + 1

SONET	VT1.5	Serial0/bay/port.<channel-id>
		<channel-id> is the channel ID calculated based on the following formula:
		Channel-id = (sts_number − 1) x 28 + (T1/E1 − 1) x 7 + VTG
		The following example describes how the channel ID is calculated for a given configuration.
		`sts-1 2`
		`mode vt-15`
		`vtg 2 t1 3 channel-group 0 timeslots 1-24`
		Inter serial interface channel-id = (2 − 1) x 28 + (3 − 1) x 7 + 2 = 44
		• VTG 1 1: T1 {1,8,15,22}
		• VTG 2 2: T1 {2,9,16,23}
		• VTG 3 3: T1 {3,10,17,24}
		• VTG 4 4: T1 {4,11,18,25}
		• VTG 5 5: T1 {5,12,19,26}
		• VTG 6 6: T1 {6,13,20,27}
		• VTG 7 7: T1 {7,14,21,28

Mode	Interface Mode	Serial Interface with supported Channel Identifier
SDH	Vc11 and Vc12	T1 number with Vc11 supported: • TUG2 1: T1 {1,8,15,22} • TUG2 2: T1 {2,9,16,23} • TUG2 3: T1 {3,10,17,24} • TUG2 4: T1 {4,11,18,25} • TUG2 5: T1 {5,12,19,26} • TUG2 6: T1 {6,13,20,27} • TUG2 7: T1 {7,14,21,28 E1 number with Vc12 supported: • TUG2 1: E1 {1,8,15} • TUG2 2: E1 {2,9,16} • TUG2 3: E1 {3,10,17} • TUG2 4: E1 {4,11,18} • TUG2 5: E1 {5,12,19} • TUG2 6: E1 {6,13,20} • TUG2 7: E1 {7,14,21} **Note** Depending on the mode selected, the number of E1 changes.

Creating T1 or E1 Serial Interfaces on T1 or E1 Ports

Creating T1 Serial Interface

To create a channel group on a T1 interface, use the following commands:

```
router(config)#controller t1 0/2/0
router(config-controller)#channel-group 0  timeslots 1-24
```

Note For T1, the channel-group ID ranges from 0 to 23.

Creating E1 Serial Interface

To create a channel group on an E1 interface, use the following commands:

```
router(config)#controller e1 0/2/0
router(config-controller)#channel-group 0  timeslots 1-31
```

Note For E1, the channel-group ID ranges from 0 to 30.

The following example explains a channel group of number 2 with time slot 1-24 is configured on the T1 interface of the controller. The default encapsulation of HDLC is used.

```
router(config)#controller t1 0/2/0
router(config-controller)#channel-group  2 timeslots  1-24
router(config-controller)#end
```

Note While specifying time slot, use the complete range, for example, 1-24 for T1 and 1-31 for E1.

The following example explains a channel group of number 10 with time slot 1-31 is configured on the E1 interface of the controller. The default encapsulation of HDLC is used.

```
router(config)#controller e1 0/3/2
router(config-controller)#channel-group 2 timeslots 1-31
router(config-controller)#end
```

Creating T3 or E3 Serial Interfaces on T3 or E3 Ports

Configuring Mode to T3 or E3

To configure T3 mode, use the following commands:

```
router(config)#controller mediatype 0/2/12
router(config-controller)#mode t3
router(config-controller)#exit
```

To configure E3 mode, use the following commands:

```
router(config)#controller mediatype 0/2/12
router(config-controller)#mode e3
router(config-controller)#exit
```

Creating T3 Serial Interface

To create a T3 interface, use the following commands:

```
router(config)#controller t3 0/2/12
router(config-controller)#no channelized
router(config-controller)#channel-group 0
router(config-controller)#exit
```

Note Use **no channel group** command to clear configured T3 channels.

Creating E3 Serial Interface

To create an E3 interface, use the following commands:

```
router(config)#controller e3 0/2/12
router(config-controller)#no channelized
router(config-controller)#channel-group 0
router(config-controller)#exit
```

Creating CT3 Serial Interface

To create a CT3 interface, use the following commands:

```
router(config)#controller t3 0/2/12
router(config-controller)#channelized
router(config-controller)#T1 1 channel-group 0 timeslots 1-24
router(config-controller)#T1 2 channel-group 0 timeslots 1-24
router(config-controller)#exit
```

Note While specifying time slot, ensure that you provide the complete time slot, for example 1-24 for T1 interface.

The following example explains a channel group of 0 is configured on the E3 interface of the controller. The default encapsulation of HDLC is used.

```
router(config)#controller e3 0/2/12
router(config-controller)#no channelized
router(config-controller)#channel-group 0
router(config-controller)#end
```

The following example explains a channel group of number 0 is configured on the CT3 interface of the controller. The default encapsulation of HDLC is used.

```
router(config)#controller t3 0/2/12
router(config-controller)#no channelized
router(config-controller)#channel-group 0
router(config-controller)#end
```

Creating Serial Interfaces on SDH

Configuring Mode to SDH

To enter into SDH mode, use the following commands:

```
router(config)#controller mediatype 0/bay/port
router(config-controller)#mode sdh
router(config-controller)#exit
```

Creating SDH T3 Interface

To create an SDH T3 interface, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#rate {stm1 | stm4 | stm16}
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#mode tug-3
router(config-ctrlr-au4)#tug-3 1
router(config-ctrlr-tug3)#[no]mode t3
router(config-ctrlr-tug3)#[no]t3 channel-group 0
router(config-ctrlr-tug3)#exit
```

Creating SDH E3 Interface

To create an SDH E3 interface, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#rate {stm1 | stm4 | stm16}
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#mode tug-3
router(config-ctrlr-au4)#tug-3 1
router(config-ctrlr-tug3)#[no]mode e3
router(config-ctrlr-tug3)#[no]e3  channel-group 0
router(config-ctrlr-tug3)#exit
```

Creating SDH VC11 Interface

To create an SDH VC11 interface, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#rate {stm1 | stm4 | stm16}
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#[no]mode tug-3
router(config-ctrlr-au4)#tug-3 1
router(config-ctrlr-tug3)#[no]mode vc1x
router(config-ctrlr-tug3)#tug-2 1 payload vc11
router(config-ctrlr-tug2-vcx)#[no]t1 1  channel-group 0 timeslots 1-24
router(config-ctrlr-tug3)#exit
```

Creating SDH VC12 Interface

To create an SDH VC12 interface, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#rate {stm1 | stm4 | stm16}
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#[no]mode tug-3
router(config-ctrlr-au4)#tug-3 1
router(config-ctrlr-tug3)#[no]mode vc1x
router(config-ctrlr-tug3)#tug-2 1 payload vc12
router(config-ctrlr-tug2-vcx)#[no]e1 1  channel-group 0 timeslots 1-31
router(config-ctrlr-tug3)#exit
```

Creating SDH VC4-nc Interface

To create an SDH VC4-nc concatenated interface, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#rate {stm1 | stm4 | stm16}
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#[no]mode vc4
router(config-ctrlr-au4)#[no]channel-group 0
router(config-ctrlr-tug3)#exit
```

Creating SDH T3 Interface with AUG-3 Mapping

To create an SDH T3 interface with AUG-3 AUG mapping, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#aug mapping au-3
router(config-controller)#au-3 1
router(config-ctrlr-au3)#[no]mode t3
router(config-ctrlr-au3)#[no]t3 channel-group 0
router(config-ctrlr-au3)#exit
```

Creating SDH VC11 Interface with AUG-3 Mapping

To create an SDH VC11 interface with AUG-3 AUG mapping, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#au-3 1
router(config-ctrlr-au3)#[no]mode vc1x
router(config-ctrlr-au3)#tug-2 1 payload vc11
router(config-ctrlr-tug2-vcx)#[no] t1 1 channel-group 0 timeslots 1-24
router(config-ctrlr-tug3)#exit
```

Creating SDH VC12 Interface with AUG-3 Mapping

To create an SDH VC12 interface with AUG-3 AUG mapping, use the following commands:

```
router(config)#controller sdh 0/bay/port
router(config-controller)#au-3 1
router(config-ctrlr-au3)#[no]mode vc1x
router(config-ctrlr-au3)#tug-2 1 payload vc12
router(config-ctrlr-tug2-vcx)#[no]e1 1 channel-group 0 timeslots 1-31
router(config-ctrlr-tug3)#exit
```

The following example explains SDH serial interface is configured with rate STM1 with AU-4 mapping and TUG-3 and T3 mode:

```
router(config)#controller sdh 0/3/4
router(config-controller)#rate stm1
router(config-controller)#aug mapping au-4
router(config-controller)#au-4 1
router(config-ctrlr-au4)#mode tug-3
router(config-ctrlr-au4)#tug-3 1
router(config-ctrlr-tug3)#mode t3
router(config-ctrlr-tug3)#t3  channel-group 0
router(config-ctrlr-tug3)#exit
```

Creating Serial Interfaces on SONET

Setting Controller Mode to SONET

To enter into SONET mode, use the following commands:

```
router(config)#controller mediatype 0/bay/port
router(config-controller)#mode sonet
router(config-controller)#exit
```

Creating T3 Serial Interface

To create a channel group on the T3 interface, use the following commands:

```
router(config)#controller sonet 0/bay/port
router(config-controller)#rate {oc3 | oc12 | oc48}
router(config-controller)#sts-1  1
router(config-controller)#[no]mode t3
router(config-controller)#[no]t3 channel-group 0
router(config-controller)#exit
```

Creating VT1.5 Serial Interface

To create a channel group on the VT1.5 interface, use the following commands:

```
router(config)#controller sonet 0/bay/port
router(config-controller)#rate oc3
router(config-controller)#sts-1  1
router(config-controller)#[no]mode vt-15
router(config-controller)#[no]vtg 1 t1 1 channel-group 0 timeslots  1-24
router(config-controller)#exit
```

Creating CT3 Serial Interface

To create a channel group on the CT3 interface, use the following commands:

```
router(config)#controller sonet 0/bay/port
router(config-controller)#rate oc3
router(config-controller)#sts-1  1
router(config-controller)#[no]mode ct3
router(config-controller)#[no]t1  1 channel-group 0  timeslots 1-24
router(config-controller)#exit
```

Note While specifying time slot, ensure that you specify the complete time slot.

Creating Concatenated Mode Serial Interface

To create a channel group on the concatenated mode serial interface, use the following commands:

```
router(config)#controller sonet 0/bay/port
router(config-controller)#rate oc3
router(config-controller)#sts-1  1 - 3 mode sts-3c
router(config-controller)#channel-group 0
router(config-controller)#exit
```

The following example explains SONET interface that is configured with OC-3 rate, STS-1 as 1, and mode as T3.The serial interface is modified for PPP encapsulation.

```
router(config)#controller sonet 0/3/4
router(config-controller)#rate oc3
router(config-controller)#sts-1  1
router(config-controller)#mode t3
router(config-controller)#t3  channel-group 0
router(config-controller)#end
router(config)#interface serial 0/3/4 .1
router(config-if)#no ip address
router(config-if)# encapsulation ppp
```

Modifying Encapsulation to PPP

By default, HDLC is used for encapsulation. You can modify encapsulation to PPP on a serial interface using the **encapsulation ppp** command.

The ***channel-id*** varies based on the mode set and the circuit type. For more information, see the Serial Interface Supported Modes section.

To modify encapsulation on the serial interface, use the following commands:

```
router(config)#interface serial 0/bay/port.channel-id
router(config-if)#no ip address
router(config-if)# encapsulation ppp
```

IPv4 Interworking Pseudowire over HDLC or PPP

L2VPN Interworking

Layer 2 transport over MPLS and IP already exists for like-to-like attachment circuits, such as Ethernet-to-Ethernet or PPP-to-PPP. Layer 2 Virtual Private Network (L2VPN) Interworking builds on this functionality by allowing disparate attachment circuits to be connected. An interworking function facilitates the translation between the different Layer 2 encapsulations.

- L2VPN Interworking Mode, on page 11
- IPv4 Interworking Pseudowire Supported Modes, on page 12

L2VPN Interworking Mode

L2VPN Interworking works in IP (routed) mode that facilitates transport of IPv4 payload in HDLC or PPP frames to Ethernet, over an MPLS network. The configuration is supported on . You specify the mode by issuing the **interworking ip** command in pseudowire-class configuration mode.

The interworking command causes the attachment circuits to be terminated locally. The **ip** keyword causes IP packets to be extracted from the attachment circuit and sent over the pseudowire. Packets with IPv4 payload only are transported over pseudowire.

IP Interworking Mode

The CE routers encapsulate the IP on the link between the CE router and PE router. A new VC type is used to signal the IP pseudowire in MPLS. Translation between the L2 and IP encapsulations across the pseudowire is required. Special consideration is given to the address resolution and routing protocol operation, because these operations are handled differently on different L2 encapsulations.

In routed interworking, IP packets that are extracted from the ACs are sent over the pseudowire. The pseudowire works in the IP Layer 2 transport (VC type 0x000B) like-to-like mode. The interworking function at the network service provider's (NSP) end performs the required adaptation that is based on the AC technology. Non-IPv4 packets are not forwarded on pseudowire. Only packets with the IPv4 payload are transported over the pseudowire.

The following table details on the packets that are terminated locally:

Table 4: List of Packets Locally Terminated

Protocol	Packets (Locally Terminated)	PID Number
Cisco HDLC	SLARP, LCP, or RARP	0x8035
Cisco HDLC	NCP or ARP	0x0806
PPP	LCP	0xCxxx to 0xFxxx
PPP	NCP	0x8xxx to 0xBxxx

HDLC or PPP to Ethernet IPv4 Interworking Pseudowire

Starting with Cisco IOS XE 16.9.1 release, the L2VPN interworking allows you to connect disparate attachment circuits, for example, TDM and Ethernet attachment circuits. The L2VPN interworking operates in IP (routed) mode that facilitates transport of IPv4 payload in HDLC or PPP frames to Ethernet, over MPLS network translation. .

For pseudowires operated in the IP (routed) mode, the IP packets are extracted from the attachment circuit and sent over the pseudowire.

Once IPv4 interworking is configured, create a serial interface with specific channel identifier.

When a serial interface is UP, an internal label is allocated and LDP negotiation with a peer is performed for a remote label. A pseudowire is created and bound to HDLC or PPP channel. Based on the pseudowire configuration, you can permit IPv4 payload traffic with an allocated internal MPLS label.

The default encapsulation for all serial interfaces is HDLC. You can change the encapsulation to PPP. You can cross connect the attachment circuit segment with specific VC identifier and the pseudowire segment.

IPv4 Interworking Pseudowire Supported Modes

IPv4 interworking pseudowire is supported on the following modes:

- T1 or E1
- T3 or E3
- Channelized T3 or E3 (channelized to T1 or E1)

- SDH
- SONET

Limitations of IPv4 Interworking Pseudowire on HDLC or PPP Serial Interfaces

The following limitations apply to IPv4 interworking pseudowire on HDLC or PPP serial interfaces:

- IPv4 interworking pseudowire with HDLC or PPP attachment circuit is supported only on the .
- IPv4 interworking is supported only for HDLC or PPP to Ethernet and not supported on MLPPP.
- L3 termination, bridging, and local switching on serial interfaces are not supported.
- Access circuit redundancy, Unidirectional Path Switched Ring (UPSR), and card protection with serial interfaces are not supported.
- IPv4 over HDLC or PPP is not supported on Nx DS0 serial interfaces.
- T1 framing SF is not supported.
- HDLC or PPP is not supported for STS-12C or VC4-4C, and STS48C or VC4-16C modes.
- HDLC or PPP is not supported for CE3 modes.

How to Configure IPv4 Interworking Pseudowire on HLDC or PPP Interface

This section provides the following information about configuring an IPv4 interworking pseudowire on an HLDC or PPP interface:

- Configuring L2VPN Interworking, on page 13
- Configuring Cross-Connect Under Attachment Circuit, on page 14

Configuring L2VPN Interworking

To configure L2VPN interworking, create a pseudowire class with the tunneling encapsulation as MPLS. The **interworking** command specifies the type of payload traffic that flows across the pseudowire tunnel. Configure pseudowire class only once on a device.

You can also configure **control-word** as an optional command.

To configure L2VPN interworking, use the following commands:

```
router>enable
router#configure terminal
router(config)#pseudowire-class pw-class-name
router(config-pw)#encapsulation mpls
router(config-pw)# interworking ip
router(config-pw)# control-word
```

Configuring Cross-Connect Under Attachment Circuit

The **xconnect** command binds the attachment circuit to an L2VPN pseudowire for cross connect service. The virtual circuit identifier creates the binding between a pseudowire that is configured on a PE router and an attachment circuit in a CE device.

To perform cross connection between an AToM routed pseudowire and attachment circuit, use the following commands:

```
router(config)#interface serial 0/bay/port.channel-id
router(config-if)#xconnect ip-address vc-id pw-class atom-iw-routed
```

Verifying IPv4 Interworking Pseudowire over HDLC or PPP Configuration

The following **show interface serial** *0/bay/port.vc-number* command displays information about encapsulation and statistics of a serial interface.

To display configuration information on the serial interface, use the **show interface serial** command:

```
Router# show interface serial 0/5/19.8
 Serial0/5/19.8 is up, line protocol is up
  Hardware is
  MTU 1500 bytes, BW 1536 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Stopped: TAGCP
  Open: IPCP, crc 16, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:04, output 00:00:04, output hang never
  Last clearing of "show interface" counters 23:52:46
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     16201 packets input, 712844 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     16205 packets output, 696835 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     1 carrier transitions
 PW stats

 0 input packets ,0 output packets,
 0 input bytes, 0 output bytes, 0 input packet drop
 no alarm present
  VC 2: timeslot(s): 1-24, Transmitter delay 0, non-inverted data
```

The **show platform software tdm-combo vc info** command helps you to identify the bay, port, STS path, T1, and channel group associated with a serial interface:

```
router#show platform software tdm-combo vc info
BAY   PORT PATH  T1      CHANNEL          VC         HWIDB
spa in bay:0 is NULL
spa in bay:1 is NULL
5     19   1      1          0        Serial0/5/19.1  1
5     19   1      8          0        Serial0/5/19.8  2
TOTAL ENTRIES :2
```

The **show running-config interface serial 0/5/19.8** command provides information on cross connect status, and local or remote label bindings:

```
router#show running-config interface serial 0/5/19.8
Building configuration...

Current configuration : 147 bytes
!
interface Serial0/5/19.8
 no ip address
 encapsulation ppp
 ppp authentication chap
 xconnect 192.0.2.6 207 encapsulation mpls pw-class ip-iw
end

BYOS-RSP3#sh xconnect all
Legend:    XC ST=Xconnect State  S1=Segment1 State  S2=Segment2 State
  UP=Up       DN=Down            AD=Admin Down      IA=Inactive
  SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware

XC ST  Segment 1                         S1 Segment 2                          S2
------+---------------------------------+--+---------------------------------+--
UP pri   ac Se0/5/19.8(PPP)              UP mpls 192.0.2.6:207                   UP
```

The **show mpls l2transport vc 207 detail** command provides information on pseudowire corresponding to VC ID 207:

```
Local interface: Se0/5/19.8 up, line protocol up, PPP up
  Interworking type is IP
  Destination address: 192.0.2.6, VC ID: 207, VC status: up
    Output interface: Gi0/3/7, imposed label stack {16}
    Preferred path: not configured
    Default path: active
    Next hop: 40.40.40.1
  Create time: 23:31:56, last status change time: 23:31:54
    Last label FSM state change time: 23:31:56
  Signaling protocol: LDP, peer 192.0.2.6:0 up
    Targeted Hello: 192.0.2.10(LDP Id) -> 192.0.2.6, LDP is UP
    Graceful restart: configured and not enabled
    Non stop routing: not configured and not enabled
    Status TLV support (local/remote)   : enabled/supported
      LDP route watch                   : enabled
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: No fault
      Last BFD dataplane     status rcvd: Not sent
      Last BFD peer monitor  status rcvd: No fault
      Last local AC  circuit status rcvd: No fault
      Last local AC  circuit status sent: No fault
      Last local PW i/f circ status rcvd: No fault
      Last local LDP TLV     status sent: No fault
      Last remote LDP TLV    status rcvd: No fault
      Last remote LDP ADJ    status rcvd: No fault
    MPLS VC labels: local 512, remote 16
    Group ID: local n/a, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  Control Word: On
  SSO Descriptor: 192.0.2.6/207, local label: 512
  Dataplane:
    SSM segment/switch IDs: 8219/8218 (used), PWID: 1
  VC statistics:
```

```
transit packet totals: receive 0, send 0
transit byte totals:   receive 0, send 0
transit packet drops:  receive 0, seq error 0, send 0
```

IPv4 Layer 3 Termination on HDLC or PPP Serial Interfaces

IPv4 Layer 3 Termination on HDLC or PPP Serial Interfaces

Starting with Cisco IOS XE 16.11.x release, you can perform IPv4 Layer 3 termination on HDLC or PPP serial interfaces on the Cisco ASR 920 Series 4-Port OC3/STM-1 or 1-Port OC12/STM-4 Module.

IPv4 routing can be performed using standard routing protocols such as OSPF, BGP, IS-IS, EIGRP, and RIP.

A maximum of 1020 serial interfaces are supported on the Cisco RSP3 module.

This feature supports MPLS IP.

Restrictions for IP4 Layer 3 Termination on HDLC or PPP Serial Interfaces

- IPv6 routing is not supported.
- ACR or UPSR protection on serial interfaces is not supported.
- Multicast and QoS features are not supported.
- Frame-relay or MLPPP is not supported.
- BFD is not supported on serial interfaces.

How to Configure IP4 Layer 3 Termination on HDLC or PPP Serial Interfaces

Configuring Protocols

Configuring Routing Protocol

You should configure routing protocols such as OSPF, BGP, IS-IS, EIGRP, and RIP.

For more information on configuring IP Routing protocols, refer the respective Guides:

https://www.cisco.com/c/en/us/support/ios-nx-os-software/ios-xe-3s/products-installation-and-configuration-guides-list.html

Configuring L3 VPN

To configure L3 VPN, refer the MPLS Virtual Private Networks chapter in the MPLS: Layer 3 VPNs Configuration Guide.

Configuring VRF

Before configuring IPv4 Layer 3 flow on a serial interface, ensure that you have configured VRF forwarding. For more information, refer Configuring VFR.

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF-lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF.

With the VRF-lite feature, the router supports multiple VPN routing or forwarding instances in customer edge devices. VRF-lite allows a service provider to support two or more VPNs with overlapping IP addresses using one interface.

To configure VRF, enter the following commands:

```
router#configure terminal
router(config)#vrf definition vrf_test
router(config-vrf)#rd 1:1
router(config-vrf)#address-family ipv4
```

Once VRF is configured, ensure that you specify the Layer 3 interface to be associated with the VRF and then associate the VRF with the Layer 3 interface using the **vrf forwarding vrf-name** command. The interface can be a routed port or SVI.

To configure VRF forwarding, enter the following commands:

```
router#configure terminal
router (config-vrf)# interface interface-id
router (config-if)#vrf forwarding vrf-name
```

Configuring IPv4 Unicast Layer 3 Termination on HDLC or PPP Interfaces

You can enable or disable IPv4 Layer 3 flow on HDLC or PPP serial interfaces. You can use the **vrf forwarding <vrf name>** command optionally on the serial interface.

You can also modify the default MTU 1500 bytes optionallyusing the **mtu** command.

To enable IPv4 Layer 3 flow on a serial interface, enter the following commands:

```
router(config)#interface serial x/y/z.channel-id
router(config-if)#vrf forwarding <vrf name> (optional)
router(config-if)#ip address <ipv4 address> <mask>
router(config-if)#mtu <bytes>
```

To disable IPv4 Layer 3 flow on a serial interface, enter the no form of the command:

```
router(config)#interface serial x/y/z.channel-id
router(config-if)#vrf forwarding <vrf name>
router(config-if)#no ip address <ipv4 address> <mask>
```

```
router(config)#interface serial x/y/z.channel-id
router(config-if)#no vrf forwarding <vrf name>
```

Verifying IPv4 Layer 3 Termination on HDLC or PPP

The following **show interface serial** *0/bay/port.vc-number* command displays information about PPP encapsulation and statistics of a serial interface.

To display configuration information on the serial interface, use the **show interface serial** command:

```
Router# show interface serial 0/5/16.1
 Serial0/5/16.1 is up, line protocol is up
  Hardware is
  Internet address is 41.41.41.1/24
  MTU 1500 bytes, BW 44210 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, crc 16, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:03, output 00:00:02, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 76000 bits/sec, 298 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     99332 packets input, 983489 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     62 packets output, 4832 bytes, 0 underruns
     0 output errors, 0 collisions, 3 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
  no alarm present
  DSU mode 0, bandwidth 0 Kbit, scramble 0, VC 3, non-inverted data
```