



Configuring Serial Interfaces

This module describes the configuration of serial interfaces.

Before you configure a serial interface, you must configure the clear channel T3/E3 controller or channelized T1/E1 controller (DS0 channel) that is associated with that interface.

Feature History for Configuring Serial Controller Interfaces

Release	Modification
Release 3.4.1	<p>This feature was introduced on the Cisco CRS-1 Router.</p> <p>Support was added on the Cisco CRS-1 Router for the following hardware:</p> <ul style="list-style-type: none">• Cisco CRS-1 SIP-800• Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA

- [Prerequisites for Configuring Serial Interfaces, on page 1](#)
- [Information About Configuring Serial Interfaces, on page 2](#)
- [How to Configure Serial Interfaces, on page 9](#)
- [Configuration Examples for Serial Interfaces, on page 25](#)

Prerequisites for Configuring Serial Interfaces

Before configuring serial interfaces, ensure that the following tasks and conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You have installed a 2-Port or 4-Port Clear Channel T3/E3 SPA.
- You should have configured the clear channel T3/E3 controller controller that is associated with the serial interface you want to configure, as described in the *Configuring Clear Channel T3/E3 Controllers* module in this manual.

Information About Configuring Serial Interfaces

To configure serial interfaces, study the following concepts:

High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs

This table provides a high-level overview of the tasks required to configure a T3 serial interface on the Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA.

Table 1: Overview: Configuring a T3 Serial Interface on a Clear Channel SPA

Step	Task	Module	Section
1.	Use the hw-module subslot command to set serial mode for the SPA to be T3, if necessary. Note By default, the 2-Port and 4-Port Clear Channel T3/E3 SPA is set to run in T3 mode.	<i>Configuring Clear Channel T3/E3 Controllers</i>	Setting the Card Type
2.	Configure the T3 controller.	<i>Configuring Clear Channel T3/E3 Controllers</i>	Setting the Card Type
3.	Configure the serial interface that is associated with the T3 controller you configured in Step 2.	<i>Configuring Serial Interfaces</i>	How to Configure Serial Interfaces

This table provides a high-level overview of the tasks required to configure an E3 serial interface on a 2-Port and 4-Port Clear Channel T3/E3 SPA.

Table 2: Overview: Configuring an E3 Serial Interface on a Clear Channel SPA

Step	Task	Module	Section
1.	Use the hw-module subslot command to set serial mode for the SPA to be E3.	<i>Configuring Clear Channel T3/E3 Controllers</i>	Setting the Card Type
2.	Configure the E3 controller.	<i>Configuring Clear Channel T3/E3 Controllers</i>	Setting the Card Type

Step	Task	Module	Section
3.	Configure the serial interface that is associated with the E3 controller you configured in Step 2.	<i>Configuring Serial Interfaces</i>	How to Configure Serial Interfaces

Cisco HDLC Encapsulation

Cisco High-Level Data Link Controller (HDLC) is the Cisco proprietary protocol for sending data over synchronous serial links using HDLC. Cisco HDLC also provides a simple control protocol called Serial Line Address Resolution Protocol (SLARP) to maintain serial link keepalives. HDLC is the default encapsulation type for serial interfaces under Cisco IOS XR software. Cisco HDLC is the default for data encapsulation at Layer 2 (data link) of the Open System Interconnection (OSI) stack for efficient packet delineation and error control.



Note Cisco HDLC is the default encapsulation type for the serial interfaces.

Cisco HDLC uses keepalives to monitor the link state, as described in the [Keepalive Timer](#).



Note Use the **debug chdlc slarp packet** command to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

PPP Encapsulation

PPP is a standard protocol used to send data over synchronous serial links. PPP also provides a Link Control Protocol (LCP) for negotiating properties of the link. LCP uses echo requests and responses to monitor the continuing availability of the link.



Note When an interface is configured with PPP encapsulation, a link is declared down, and full LCP negotiation is re-initiated after five ECHOREQ packets are sent without receiving an ECHOREP response.

PPP provides the following Network Control Protocols (NCPs) for negotiating properties of data protocols that will run on the link:

- IP Control Protocol (IPCP) to negotiate IP properties
- Multiprotocol Label Switching control processor (MPLSCP) to negotiate MPLS properties
- Cisco Discovery Protocol control processor (CDPCP) to negotiate CDP properties
- IPv6CP to negotiate IP Version 6 (IPv6) properties
- Open Systems Interconnection control processor (OSICP) to negotiate OSI properties

PPP uses keepalives to monitor the link state, as described in the [Keepalive Timer](#).

PPP supports the following authentication protocols, which require a remote device to prove its identity before allowing data traffic to flow over a connection:

- Challenge Handshake Authentication Protocol (CHAP)—CHAP authentication sends a challenge message to the remote device. The remote device encrypts the challenge value with a shared secret and returns the encrypted value and its name to the local router in a response message. The local router attempts to match the name of the remote device with an associated secret stored in the local username or remote security server database; it uses the stored secret to encrypt the original challenge and verify that the encrypted values match.
- Microsoft Challenge Handshake Authentication Protocol (MS-CHAP)—MS-CHAP is the Microsoft version of CHAP. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a personal computer using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server.
- Password Authentication Protocol (PAP)—PAP authentication requires the remote device to send a name and a password, which are checked against a matching entry in the local username database or in the remote security server database.



Note For more information on enabling and configuring PPP authentication protocols, see the *Configuring PPP* module in this manual.

Use the **ppp authentication** command in interface configuration mode to enable CHAP, MS-CHAP, and PAP on a serial interface.



Note Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.

Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) is supported on these SPAs:

MLPPP provides a method for combining multiple physical links into one logical link. The implementation of MLPPP combines multiple PPP serial interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links.

MLPPP provides the same features that are supported on PPP Serial interfaces with the exception of QoS. It also provides the following additional features:

- Fragment sizes of 128, 256, and 512 bytes
- Long sequence numbers (24-bit)
- Lost fragment detection timeout period of 80 ms
- Minimum-active-links configuration option
- LCP echo request/reply support over multilink interface
- Full T1 and E1 framed and unframed links

For more information about configuring MLPPP on a serial interface, see the *Configuring PPP* module in this document.

Keepalive Timer

Cisco keepalives are useful for monitoring the link state. Periodic keepalives are sent to and received from the peer at a frequency determined by the value of the keepalive timer. If an acceptable keepalive response is not received from the peer, the link makes the transition to the down state. As soon as an acceptable keepalive response is obtained from the peer or if keepalives are disabled, the link makes the transition to the up state.



Note The **keepalive** command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation.

For each encapsulation type, a certain number of keepalives ignored by a peer triggers the serial interface to transition to the down state. For HDLC encapsulation, three ignored keepalives causes the interface to be brought down. For PPP encapsulation, five ignored keepalives causes the interface to be brought down. ECHOREQ packets are sent out only when LCP negotiation is complete (for example, when LCP is open).



Note Use the **keepalive** command in interface configuration mode to set the frequency at which LCP sends ECHOREQ packets to its peer. To restore the system to the default keepalive interval of 10 seconds, use the **keepalive** command with **no** argument. To disable keepalives, use the **keepalive disable** command. For both PPP and Cisco HDLC, a keepalive of 0 disables keepalives and is reported in the **show running-config** command output as **keepalive disable**.



Note Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend configuring a keepalive interval of 10 seconds or more on a Cisco CRS-1 Router.

When LCP is running on the peer and receives an ECHOREQ packet, it responds with an echo reply (ECHOREP) packet, regardless of whether keepalives are enabled on the peer.

Keepalives are independent between the two peers. One peer end can have keepalives enabled; the other end can have them disabled. Even if keepalives are disabled locally, LCP still responds with ECHOREP packets to the ECHOREQ packets it receives. Similarly, LCP also works if the period of keepalives at each end is different.



Note Use the **debug chdlc slarp packet** command and other Cisco HDLC **debug** commands to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

Frame Relay Encapsulation

When Frame Relay encapsulation is enabled on a serial interface, the interface configuration is hierarchical and comprises the following elements:

1. The serial main interface comprises the physical interface and port. If you are not using the serial interface to support Cisco HDLC and PPP encapsulated connections, then you must configure subinterfaces with permanent virtual circuits (PVCs) under the serial main interface. Frame Relay connections are supported on PVCs only.
2. Serial subinterfaces are configured under the serial main interface. A serial subinterface does not actively carry traffic until you configure a PVC under the serial subinterface. Layer 3 configuration typically takes place on the subinterface.
3. Point-to-point PVCs are configured under a serial subinterface. You cannot configure a PVC directly under a main interface. A single point-to-point PVC is allowed per subinterface. PVCs use a predefined circuit path and fail if the path is interrupted. PVCs remain active until the circuit is removed from either configuration. Connections on the serial PVC support Frame Relay encapsulation only.



Note The administrative state of a parent interface drives the state of the subinterface and its PVC. When the administrative state of a parent interface or subinterface changes, so does the administrative state of any child PVC configured under that parent interface or subinterface.

To configure Frame Relay encapsulation on serial interfaces, use the **encapsulation frame-relay** command.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (default)
- IETF

Use the **encap** command in PVC configuration mode to configure Cisco or IETF encapsulation on a PVC. If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.



Note Cisco encapsulation is required on serial main interfaces that are configured for MPLS. IETF encapsulation is not supported for MPLS.

Before you configure Frame Relay encapsulation on an interface, you must verify that all prior Layer 3 configuration is removed from that interface. For example, you must ensure that there is no IP address configured directly under the main interface; otherwise, any Frame Relay configuration done under the main interface will not be viable.

LMI on Frame Relay Interfaces

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay UNI interface. By default, **cisco** LMI is enabled on all PVCs. However, you can modify the default LMI type to be ANSI or Q.933.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA:

$$(MTU - 13) / 8 = \text{maximum number of PVCs}$$



Note The default setting of the **mtu** command for a serial interface is 1504 bytes. Therefore, the default numbers of PVCs supported on a serial interface configured with **cisco** LMI is 186.

Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay

The Layer 2 Tunnel Protocol Version 3 (L2TPv3) feature defines the L2TP protocol for tunneling Layer 2 payloads over an IP core network using Layer 2 virtual private networks (VPNs).

L2TPv3 is a tunneling protocol used for transporting Layer 2 protocols. It can operate in a number of different configurations and tunnel a number of different Layer 2 protocols and connections over a packet-switched network.

Before you can configure L2TPv3, you need to configure a connection between the two attachment circuits (ACs) that will host the L2TPv3 pseudowire. Cisco IOS XR software supports a point-to-point, end-to-end service, where two ACs are connected together.

This module describes how to configure a Layer 2 AC on a Frame Relay encapsulated serial interface.



Note Serial interfaces support DLCI mode layer 2 ACs only; layer 2 port mode ACs are not supported on serial interfaces.

For detailed information about configuring L2TPv3 in your network, see the “*Implementing Layer 2 Tunnel Protocol Version 3*” module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router*. For detailed information about configuring L2VPNs, see the “*Implementing MPLS Layer 2 VPNs*” module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router*.

Default Settings for Serial Interface Configurations

When an interface is enabled on a T3/E3 SPA, and no additional configuration commands are applied, the default interface settings shown in this table are present. These default settings can be changed by configuration.

Table 3: Serial Interface Default Settings

Parameter	Configuration File Entry	Default Settings
Keepalive Note The keepalive command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation.	keepalive [disable] no keepalive	keepalive 10 seconds
Encapsulation	encapsulation [hdlc ppp frame-relay [ietf]]	hdlc
Maximum transmission unit (MTU)	mtu bytes	1504 bytes
Cyclic redundancy check (CRC)	crc [16 32]	16
Data stream inversion on a serial interface	invert	Data stream is not inverted
Payload scrambling (encryption)	scramble	Scrambling is disabled.
Number of High-Level Data Link Control (HDLC) flag sequences to be inserted between the packets	transmit-delay	Default is 0 (disabled).



Note Default settings do not appear in the output of the show running-config command.

Serial Interface Naming Notation

The naming notation for serial interfaces on a clear channel SPA is *rack/slot/module/port*, as shown in this example:

```
interface serial 0/0/1/2
```

The naming notation for T1, E1, and DS0 interfaces on a channelized SPA is *rack/slot/module/port/channel-num:channel-group-number*, as shown in the following example:

```
interface serial 0/0/1/2/4:3
```

If a subinterface and PVC are configured under the serial interface, then the router includes the subinterface number at the end of the serial interface address. In this case, the naming notation is *rack/slot/module/port[/channel-num:channel-group-number].subinterface*, as shown in the following examples:

```
interface serial 0/0/1/2.1
interface serial 0/0/1/2/4:3.1
```



Note A slash between values is required as part of the notation.

The naming notation syntax for serial interfaces is as follows:

- *rack*: Chassis number of the rack.
- *slot*: Physical slot number of the modular services card or line card.
- *module*: Module number. Shared port adapters (SPAs) are referenced by their subslot number.
- *port*: Physical port number of the controller.
- *channel-num*: T1 or E1 channel number. T1 channels range from 0 to 23; E1 channels range from 0 to 31.
- *channel-group-number*: Time slot number. T1 time slots range from 1 to 24; E1 time slots range from 1 to 31. The *channel-group-number* is preceded by a colon and not a slash.
- *subinterface*: Subinterface number.

Use the question mark (?) online help function following the **serial** keyword to view a list of all valid interface choices.

How to Configure Serial Interfaces

After you have configured a channelized or clear channel T3/E3 controller, as described in the *Configuring Clear Channel T3/E3 Controllers* module in this document, you can configure the serial interfaces associated with that controller.

Bringing Up a Serial Interface

This task describes the commands used to bring up a serial interface.

Before you begin

- The Cisco CRS-1 Router must have the following SIP and SPA installed and running Cisco IOS XR software:
- Cisco CRS-1 SIP-800
- 2-Port and 4-Port T3/E3 Serial SPA

Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **interface serial** *interface-path-id*
4. **ipv4 address ip-address**
5. **no shutdown**
6. **end** or **commit**
7. **exit**
8. **exit**
9. Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
10. **show ipv4 interface brief**
11. **show interfaces serial** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	show interfaces Example: RP/0/RP0/CPU0:router# show interfaces	(Optional) Displays configured interfaces. <ul style="list-style-type: none"> • Use this command to also confirm that the router recognizes the PLIM card.
Step 2	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 3	interface serial <i>interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0	Specifies the serial interface name and notation <i>rack/slot/module/port</i> , and enters interface configuration mode.
Step 4	ipv4 address ip-address Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.1.2.1 255.255.255.224	Assigns an IP address and subnet mask to the interface. Note <ul style="list-style-type: none"> • Skip this step if you are configuring Frame Relay encapsulation on this interface. For Frame Relay, the IP address and subnet mask are configured under the subinterface.
Step 5	no shutdown Example: RP/0/RP0/CPU0:router (config-if)# no shutdown	Removes the shutdown configuration. Note <ul style="list-style-type: none"> • Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming the parent SONET layer is not configured administratively down).

	Command or Action	Purpose
Step 6	<p>end or commit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-if)# end</pre> <p>OR</p> <pre>RP/0/RP0/CPU0:router (config-if)# commit</pre>	<p>Saves configuration changes.</p> <ul style="list-style-type: none"> When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-if)# exit</pre>	Exits interface configuration mode and enters global configuration mode.
Step 8	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config)# exit</pre>	Exits global configuration mode and enters EXEC mode.
Step 9	<p>Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show interfaces</pre> <pre>RP/0/RP0/CPU0:router# configure RP/0/RP0/CPU0:router (config)# interface serial 0/1/0/1 RP/0/RP0/CPU0:router (config-if)# ipv4 address 10.1.2.2 255.255.255.224 RP/0/RP0/CPU0:router (config-if)# no shutdown RP/0/RP0/CPU0:router (config-if)# commit RP/0/RP0/CPU0:router (config-if)# exit RP/0/RP0/CPU0:router (config)# exit</pre>	<p>Note</p> <ul style="list-style-type: none"> The configuration on both ends of the serial connection must match.
Step 10	show ipv4 interface brief	Verifies that the interface is active and properly configured.

	Command or Action	Purpose
	Example: <pre>RP/0/RP0/CPU0:router # show ipv4 interface brief</pre>	If you have brought up a serial interface properly, the “Status” field for that interface in the show ipv4 interface brief command output displays “Up.”
Step 11	show interfaces serial <i>interface-path-id</i> Example: <pre>RP/0/RP0/CPU0:router# show interfaces serial 0/1/0/0</pre>	(Optional) Displays the interface configuration.

What to do next

To modify the default configuration of the serial interface you just brought up, see the “Configuring Optional Serial Interface Parameters” section on page 564.

Configuring Optional Serial Interface Parameters

This task describes the commands used to modify the default configuration on a serial interface.

Before you begin

Before you modify the default serial interface configuration, you must bring up the serial interface and remove the shutdown configuration, as described in the [Bringing Up a Serial Interface](#).

Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **encapsulation** [hdlc | ppp | frame-relay [IETF]
4. **serial**
5. **crc length**
6. **invert**
7. **scramble**
8. **transmit-delay** *hdlc-flags*
9. **end** or **commit**
10. **exit**
11. **exit**
12. **exit**
13. **show interfaces serial** [*interface-path-id*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0	Specifies the serial interface name and notation <i>rack/slot/module/port</i> , and enters interface configuration mode.
Step 3	encapsulation [hdlc ppp frame-relay [IETF]] Example: RP/0/RP0/CPU0:router(config-if)# encapsulation hdlc	(Optional) Configures the interface encapsulation parameters and details such as HDLC, PPP or Frame Relay. Note • The default encapsulation is hdlc .
Step 4	serial Example: RP/0/RP0/CPU0:router(config-if)# serial	(Optional) Enters serial submode to configure the serial parameters.
Step 5	crc <i>length</i> Example: RP/0/RP0/CPU0:ios(config-if-serial)# crc 32	(Optional) Specifies the length of the cyclic redundancy check (CRC) for the interface. Enter the 16 keyword to specify 16-bit CRC mode, or enter the 32 keyword to specify 32-bit CRC mode. Note • The default is CRC length is 16.
Step 6	invert Example: RP/0/RP0/CPU0:ios(config-if-serial)# inverts	(Optional) Inverts the data stream.
Step 7	scramble Example: RP/0/RP0/CPU0:ios(config-if-serial)# scramble	(Optional) Enables payload scrambling on the interface. Note • Payload scrambling is disabled on the interface.
Step 8	transmit-delay <i>hdlc-flags</i> Example: RP/0/RP0/CPU0:ios(config-if-serial)# transmit-delay 10	(Optional) Specifies a transmit delay on the interface. Values can be from 0 to 128. Note • Transmit delay is disabled by default (the transmit delay is set to 0).
Step 9	end or commit	Saves configuration changes.

	Command or Action	Purpose
	<p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-if)# end</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config-if)# commit</pre>	<ul style="list-style-type: none"> When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 10	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-if-serial)# exit</pre>	Exits serial configuration mode.
Step 11	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-if)# exit</pre>	Exits interface configuration mode and enters global configuration mode.
Step 12	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config)# exit</pre>	Exits global configuration mode and enters EXEC mode.
Step 13	<p>show interfaces serial [<i>interface-path-id</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show interface serial 0/1/0/0</pre>	(Optional) Displays general information for the specified serial interface.

What to do next

- To create a point-to-point Frame Relay subinterface with a PVC on the serial interface you just brought up, see the “Creating a Point-to-Point Serial Subinterface with a PVC” section on page 567.
- To configure PPP authentication on serial interfaces with PPP encapsulation, see the “Configuring PPP on the Cisco ASR 9000 Series Router” module later in this manual.

- To modify the default keepalive configuration, see the “Modifying the Keepalive Interval on Serial Interfaces” section on page 572.

Creating a Point-to-Point Serial Subinterface with a PVC

The procedure in this section creates a point-to-point serial subinterface and configures a permanent virtual circuit (PVC) on that serial subinterface.



Note Subinterface and PVC creation is supported on interfaces with Frame Relay encapsulation only.

Before you begin

Before you can create a subinterface on a serial interface, you must bring up the main serial interface with Frame Relay encapsulation, as described in the [Bringing Up a Serial Interface](#).

Restrictions

Only one PVC can be configured for each point-to-point serial subinterface.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *dci*
5. **end** or **commit**
6. Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id.subinterface</i> point-to-point Example: RP/0/RP0/CPU0:router (config)# interface serial 0/1/0/0.1	Enters serial subinterface configuration mode.
Step 3	ipv4 address <i>ipv4_address/prefix</i> Example:	Assigns an IP address and subnet mask to the subinterface.

	Command or Action	Purpose
	RP/0/RP0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24	
Step 4	<p>pvc dlc</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-subif)# pvc 20</pre>	<p>Creates a serial permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.</p> <p>Replace <i>dlci</i> with a PVC identifier, in the range from 16 to 1007.</p> <p>Note</p> <ul style="list-style-type: none"> • Only one PVC is allowed per subinterface.
Step 5	<p>end or commit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-subif)# end</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router (config-subif)# commit</pre>	<p>Saves configuration changes.</p> <ul style="list-style-type: none"> • When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> - Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. - Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. - Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 6	<p>Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# configure RP/0/RP0/CPU0:router (config)# interface serial 0/1/0/1.1 RP/0/RP0/CPU0:router (config-subif)#ipv4 address 10.46.8.5/24 RP/0/RP0/CPU0:router (config-subif)# pvc 20 RP/0/RP0/CPU0:router (config-fr-vc)# commit</pre>	<p>Note</p> <ul style="list-style-type: none"> • The DLCI (or PVC identifier) must match on both ends of the subinterface connection. • When assigning an IP address and subnet mask to the subinterface at the other end of the connection, keep in mind that the addresses at both ends of the connection must be in the same subnet.

What to do next

- To configure optional PVC parameters, see the “Configuring Optional Serial Interface Parameters” section on page 564.
- To attach a Layer 3 QOS service policy to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.

Configuring Optional PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial PVC.

Before you begin

Before you can modify the default PVC configuration, you must create the PVC on a serial subinterface, as described in the [Creating a Point-to-Point Serial Subinterface with a PVC](#).

Restrictions

- The DLCI (or PVI identifier) must match on both ends of the PVC for the connection to be active.
- To change the PVC DLCI, you must delete the PVC and then add it back with the new DLCI.

SUMMARY STEPS

1. Configuring Optional PVC Parameters
2. **interface serial** *interface-path-id.subinterface*
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end** or **commit**
7. Repeat Step 1 through Step 6 to bring up the serial subinterface and any associated PVC at the other end of the connection.
8. **show frame-relay pvc** *dlci-number*
9. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**} or **show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Configuring Optional PVC Parameters Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id.subinterface</i> Example: RP/0/RP0/CPU0:router (config)# interface serial 0/1/0/0.1	Enters serial subinterface configuration mode.

	Command or Action	Purpose
Step 3	<p>pvc dlcI</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-subif)# pvc 20</pre>	Enters subinterface configuration mode for the PVC.
Step 4	<p>encap [cisco ietf]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-fr-vc)# encap ietf</pre>	<p>(Optional) Configures the encapsulation for a Frame Relay PVC.</p> <p>Note</p> <ul style="list-style-type: none"> If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.
Step 5	<p>service-policy {input output} policy-map</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-fr-vc)# service-policy output policy1</pre>	Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.
Step 6	<p>end or commit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router (config-fr-vc)# end</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router (config-fr-vc)# commit</pre>	<p>Saves configuration changes.</p> <ul style="list-style-type: none"> When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	<p>Repeat Step 1 through Step 6 to bring up the serial subinterface and any associated PVC at the other end of the connection.</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# configure</pre>	<p>Note</p> <ul style="list-style-type: none"> The configuration on both ends of the subinterface connection must match.

	Command or Action	Purpose
	<pre>RP/0/RP0/CPU0:router (config)# interface serial 0/1/0/1.1 RP/0/RP0/CPU0:router (config-subif)# pvc 20 RP/0/RP0/CPU0:router (config-fr-vc)# encaps cisco RP/0/RP0/CPU0:router (config-fr-vc)# commit</pre>	
Step 8	<p>show frame-relay pvc dcli-number</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show frame-relay pvc 20</pre>	(Optional) Verifies the configuration of specified serial interface.
Step 9	<p>show policy-map interface serial interface-path-id.subinterface {input output} or show policy-map type qos interface serial interface-path-id.subinterface {input output}</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show policy-map interface serial 0/1/0/0.1 output or RP/0/RP0/CPU0:router# show policy-map type qos interface serial 0/1/0/0.1 output</pre>	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

Modifying the Keepalive Interval on Serial Interfaces

Perform this task to modify the keepalive interval on serial interfaces that have Cisco HDLC or PPP encapsulation enabled.



Note When you enable Cisco HDLC or PPP encapsulation on a serial interface, the default keepalive interval is 10 seconds. Use this procedure to modify that default keepalive interval.

Cisco HDLC is enabled by default on serial interfaces.

Before you begin

Before modifying the keepalive timer configuration, ensure that Cisco HDLC or PPP encapsulation is enabled on the interface. Use the **encapsulation** command to enable Cisco HDLC or PPP encapsulation on the interface, as described in the [Configuring Optional Serial Interface Parameters](#).

Restrictions

- Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend configuring a keepalive interval of 10 seconds or more on a Cisco CRS-1 Router.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **keepalive** {*seconds* | **disable**} or **no keepalive**
4. **end** or **commit**
5. **show interfaces serial** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0	Specifies the serial interface name and notation <i>rack/slot/module/port</i> and enters interface configuration mode.
Step 3	keepalive { <i>seconds</i> disable } or no keepalive Example: RP/0/RP0/CPU0:router(config-if)# keepalive 3 or RP/0/RP0/CPU0:router(config-if)# no keepalive	Specifies the number of seconds between keepalive messages. <ul style="list-style-type: none"> • Use the keepalive disable command, the no keepalive, or the keepalive command with an argument of 0 to disable the keepalive feature. • The range is from 1 to 30 seconds. The default is 10 seconds. • If keepalives are configured on an interface, use the no keepalive command to disable the keepalive feature before configuring Frame Relay encapsulation on that interface.
Step 4	end or commit Example: RP/0/RP0/CPU0:router(config-if)# end or RP/0/RP0/CPU0:router(config-if)# commit	Saves configuration changes. <ul style="list-style-type: none"> • When you issue the end command, the system prompts you to commit changes: Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: - Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. - Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.

	Command or Action	Purpose
		<ul style="list-style-type: none"> - Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	<p>show interfaces serial <i>interface-path-id</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show interfaces serial 0/1/0/0</pre>	(Optional) Verifies the interface configuration.

How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- [Creating a Serial Layer 2 Subinterface with a PVC](#)
- [Configuring Optional Serial Layer 2 PVC Parameters](#)



Note After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible. If any routing commands are configured on the interface, then the **l2transport** command is rejected.

Creating a Serial Layer 2 Subinterface with a PVC

The procedure in this section creates a Layer 2 subinterface with a PVC.

Before you begin

Before you can create a subinterface on a serial interface, you must bring up a serial interface, as described in the [Bringing Up a Serial Interface](#).

Restrictions

Only one PVC can be configured for each serial subinterface.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **end** or **commit**
5. Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id.subinterface</i> l2transport Example: RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport	Creates a subinterface and enters serial subinterface configuration mode for that subinterface.
Step 3	pvc <i>vpi/vci</i> Example: RP/0/RP0/CPU0:router(config-if)# pvc 5/20	Creates a serial permanent virtual circuit (PVC) and enters serial Layer 2 transport PVC configuration mode. Note <ul style="list-style-type: none"> • Only one PVC is allowed per subinterface.
Step 4	end or commit Example: RP/0/RP0/CPU0:router(config-fr-vc)# end or RP/0/RP0/CPU0:router(config-fr-vc)# commit	Saves configuration changes. <ul style="list-style-type: none"> • When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> - Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. - Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. - Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC.	Brings up the AC. Note <ul style="list-style-type: none"> • The configuration on both ends of the AC must match.

What to do next

- To configure optional PVC parameters, see the “Configuring Optional Serial Layer 2 PVC Parameters” section on page 577.
- For detailed information about configuring L2TPv3 in your network, see the “Implementing Layer 2 Tunnel Protocol Version 3” module of the Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router. For detailed information about configuring L2VPNs, see the “Implementing MPLS Layer 2 VPNs” module of the Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.

Configuring Optional Serial Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial Layer 2 PVC.

Before you begin

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 subinterface, as described in the [Creating a Serial Layer 2 Subinterface with a PVC](#).

Restrictions

The configuration on both ends of the PVC must match for the connection to be active.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** {**cisco** | **ietf**}
5. **service-policy** {**input** | **output**} *policy-map*
6. **end** or **commit**
7. Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.
8. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**} or **show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	interface serial <i>interface-path-id.subinterface</i> l2transport Example: RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport	Enters serial subinterface configuration mode for a Layer 2 serial subinterface.

	Command or Action	Purpose
Step 3	<p>pvc dlc</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-if)# pvc 100</pre>	Enters serial Frame Relay PVC configuration mode for the specified PVC.
Step 4	<p>encap {cisco ietf}</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-fr-vc)# encapsulation aal5</pre>	Configures the encapsulation for a Frame Relay PVC.
Step 5	<p>service-policy {input output} policy-map</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-subif)# service-policy output policy1</pre>	Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.
Step 6	<p>end or commit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-serial-l2transport-pvc)# end</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config-serial-l2transport-pvc)# commit</pre>	<p>Saves configuration changes.</p> <ul style="list-style-type: none"> When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session. <p>Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</p>
Step 7	Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.	<p>Brings up the AC.</p> <p>Note</p> <ul style="list-style-type: none"> The configuration on both ends of the connection must match.

	Command or Action	Purpose
Step 8	<p>show policy-map interface serial <i>interface-path-id.subinterface</i> {input output} or show policy-map type qos interface serial <i>interface-path-id.subinterface</i> {input output}</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show policy-map interface pos 0/1/0/0.1 output</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router# show policy-map type qos interface pos 0/1/0/0.1 output</pre>	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

What to do next

- To configure a point-to-point pseudowire XConnect on the AC you just created, see the “Implementing Layer 2 Tunnel Protocol Version 3” module of the Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.
- To configure an L2VPN, see the “Implementing MPLS Layer 2 VPNs” module of the Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.

Configuration Examples for Serial Interfaces

This section provides the following configuration examples:

Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example

The following example shows how to bring up a basic serial interface with Cisco HDLC encapsulation:

```
RP/0/RP0/CPU0:Router#config
RP/0/RP0/CPU0:Router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:Router(config-if)# ipv4 address 192.0.2.2 255.255.255.252
RP/0/RP0/CPU0:Router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure the interval between keepalive messages to be 10 seconds:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# keepalive 10
RP/0/RP0/CPU0:router(config-if)# commit
```

The following example shows how to modify the optional serial interface parameters:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0:0
RP/0/RP0/CPU0:Router(config-if)# serial
RP/0/RP0/CPU0:Router(config-if-serial)# crc 16
RP/0/RP0/CPU0:Router(config-if-serial)# invert
RP/0/RP0/CPU0:Router(config-if-serial)# scramble
RP/0/RP0/CPU0:Router(config-if-serial)# transmit-delay 3
RP/0/RP0/CPU0:Router(config-if-serial)# commit
```

The following is sample output from the **show interfaces serial** command:

```
RP/0/RP0/CPU0:Router# show interfaces serial 0/0/3/0/5:23
Serial0/0/3/0/5:23 is down, line protocol is down
  Hardware is Serial network interface(s)
  Internet address is Unknown
  MTU 1504 bytes, BW 64 Kbit
    reliability 143/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, crc 16, loopback not set, keepalive set (10 sec)
  Last clearing of "show interface" counters 18:11:15
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
    2764 packets input, 2816 bytes, 3046 total input drops
    0 drops for unrecognized upper-level protocol
  Received 0 broadcast packets, 0 multicast packets
    0 runts, 0 giants, 0 throttles, 0 parity
  3046 input errors, 1 CRC, 0 frame, 0 overrun, 2764 ignored, 281 abort
  2764 packets output, 60804 bytes, 0 total output drops
  Output 0 broadcast packets, 0 multicast packets
  0 output errors, 0 underruns, 0 applique, 0 resets
  0 output buffer failures, 0 output buffers swapped out
  0 carrier transitions
```

Configuring a Serial Interface with Frame Relay Encapsulation: Example

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 1:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0
RP/0/RP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RP0/CPU0:router(config-if)# frame-relay intf-type dce
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/0.1 point-to-point
RP/0/RP0/CPU0:router (config-subif)# ipv4 address 10.20.3.1/24

RP/0/RP0/CPU0:router (config-subif)# pvc 16

RP/0/RP0/CPU0:router (config-fr-vc)# encapsulation ietf

RP/0/RP0/CPU0:router (config-fr-vc)# commit
```

```

RP/0/RP0/CPU0:router(config-fr-vc)# exit

RP/0/RP0/CPU0:router(config-subif)# exit

RP/0/RP0/CPU0:router(config)# exit

RP/0/RP0/CPU0:router# show interface serial 0/1/0/0
Wed Oct  8 04:14:39.946 PST DST
Serial0/1/0/0 is up, line protocol is up
  Interface state transitions: 5
  Hardware is Serial network interface(s)
  Internet address is 10.20.3.1/24
  MTU 4474 bytes, BW 44210 Kbit
    reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 16,
  Scrambling is disabled, Invert data is disabled
  LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
  LMI enq recvd 880, LMI stat sent  880, LMI upd sent  0 , DCE LMI up
  LMI DLCI 1023  LMI type is CISCO frame relay DCE
  Last clearing of "show interface" counters 02:23:04
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
    858 packets input, 11154 bytes, 0 total input drops
    0 drops for unrecognized upper-level protocol
    Received 0 runts, 0 giants, 0 throttles, 0 parity
    0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
    858 packets output, 12226 bytes, 0 total output drops
    0 output errors, 0 underruns, 0 applique, 0 resets
    0 output buffer failures, 0 output buffers swapped out

```

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 2, which is connected to router 1:

```

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/1
RP/0/RP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/1/0/1.1 point-to-point
RP/0/RP0/CPU0:router (config-subif)#ipv4 address 10.20.3.2/24

RP/0/RP0/CPU0:router (config-subif)# pvc 16

RP/0/RP0/CPU0:router (config-fr-vc)# encapsulation ietf

RP/0/RP0/CPU0:router (config-fr-vc)# commit

RP/0/RP0/CPU0:router(config-fr-vc)# exit

RP/0/RP0/CPU0:router(config-subif)# exit

```

```

RP/0/RP0/CPU0:router(config)# exit

RP/0/RP0/CPU0:router# show interface serial 0/1/0/1
Wed Oct  8 04:13:45.046 PST DST
Serial0/1/0/1 is up, line protocol is up
Interface state transitions: 7
Hardware is Serial network interface(s)
Internet address is Unknown
MTU 4474 bytes, BW 44210 Kbit
    reliability 255/255, txload 0/255, rxload 0/255
Encapsulation FRAME-RELAY, crc 16,
Scrambling is disabled, Invert data is disabled
LMI enq sent 1110, LMI stat recvd 875, LMI upd recvd 0, DTE LMI up
LMI enq recvd 0, LMI stat sent 0, LMI upd sent 0
LMI DLCI 1023 LMI type is CISCO frame relay DTE
Last clearing of "show interface" counters 02:22:09
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  853 packets input, 12153 bytes, 0 total input drops
  0 drops for unrecognized upper-level protocol
  Received 0 runts, 0 giants, 0 throttles, 0 parity
  0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
  853 packets output, 11089 bytes, 0 total output drops
  0 output errors, 0 underruns, 0 applique, 0 resets
  0 output buffer failures, 0 output buffers swapped out

```

Configuring a Serial Interface with PPP Encapsulation: Example

The following example shows how to create and configure a serial interface with PPP encapsulation:

```

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# ppp authentication chap MIS-access
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

```

The following example shows how to configure serial interface 0/3/0/0/0:0 to allow two additional retries after an initial authentication failure (for a total of three failed authentication attempts):

```

RP/0/RP0/CPU0:router# configuration
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# ppp authentication chap
RP/0/RP0/CPU0:router(config-if)# ppp max-bad-auth 3
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

```