MPLS Transport Profile

Note This chapter is not applicable on the ASR 900 RSP3 Module.

Multiprotocol Label Switching (MPLS) Transport Profile (TP) enables you to create tunnels that provide the transport network service layer over which IP and MPLS traffic traverses. MPLS-TP tunnels enable a transition from Synchronous Optical Networking (SONET) and Synchronous Digital Hierarchy (SDH) time-division multiplexing (TDM) technologies to packet switching to support services with high bandwidth requirements, such as video.

- Restrictions for MPLS-TP on the Cisco ASR 900 Series Routers, on page 1
- Information About MPLS-TP, on page 2
- How to Configure MPLS Transport Profile, on page 8
- Configuration Examples for MPLS Transport Profile, on page 28

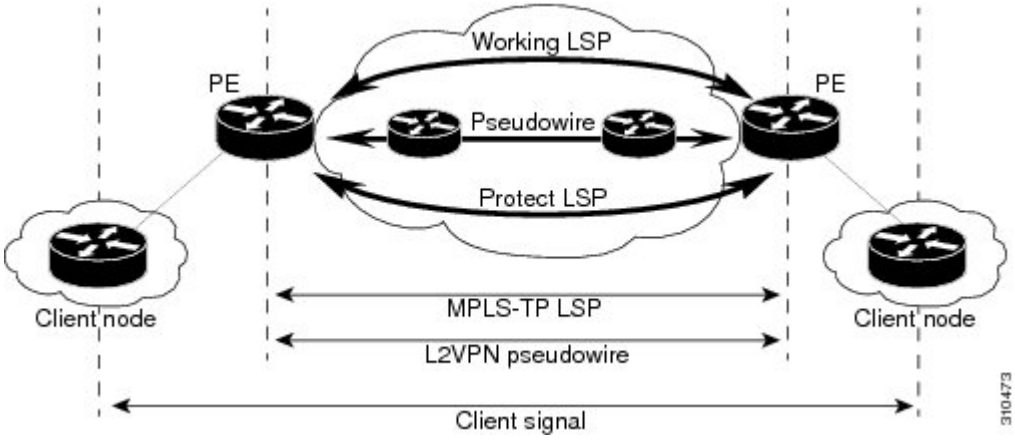
Restrictions for MPLS-TP on the Cisco ASR 900 Series Routers

- Multiprotocol Label Switching Transport Profile (MPLS-TP) penultimate hop popping is *not* supported. Only ultimate hop popping is supported, because label mappings are configured at the MPLS-TP endpoints
- IPv6 addressing is *not* supported.
- VCCV BFD is *not* supported.
- Layer 2 Virtual Private Network (L2VPN) interworking is *not* supported.
- Local switching with Any Transport over MPLS (AToM) pseudowire as a backup is *not* supported.
- L2VPN pseudowire redundancy to an AToM pseudowire by one or more attachment circuits is *not* supported.
- Pseudowire ID Forward Equivalence Class (FEC) type 128 is supported, but generalized ID FEC type 129 is *not* supported
- Maximum virtual circuits (VC) supported for MPLS-TP is 2000.

Information About MPLS-TP

How MPLS Transport Profile Works

Multiprotocol Label Switching Transport Profile (MPLS-TP) tunnels provide the transport network service layer over which IP and MPLS traffic traverses. MPLS-TP tunnels help transition from Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) and Time Division Multiplexing (TDM) technologies to packet switching to support services with high bandwidth utilization and lower cost. Transport networks are connection-oriented, statically provisioned, and have long-lived connections. Transport networks usually avoid control protocols that change identifiers (like labels). MPLS-TP tunnels provide this functionality through statically provisioned bidirectional label switched paths (LSPs), as shown in the figure below.



MPLS-TP is supported on ATM and TDM pseudowires on the Cisco ASR 903 router. For information, see Configuring the Pseudowire Class.

MPLS-TP Path Protection

MPLS-TP label switched paths (LSPs) support 1-to-1 path protection. There are two types of LSPs: protect LSPs and working LSPs. You can configure the both types of LSPs when configuring the MPLS-TP tunnel. The working LSP is the primary LSP used to route traffic. The protect LSP acts as a backup for a working LSP. If the working LSP fails, traffic is switched to the protect LSP until the working LSP is restored, at which time forwarding reverts back to the working LSP.

Bidirectional LSPs

Multiprotocol Label Switching Transport Profile (MPLS-TP) label switched paths (LSPs) are bidirectional and co-routed. They comprise of two unidirectional LSPs that are supported by the MPLS forwarding infrastructure. A TP tunnel consists of a pair of unidirectional tunnels that provide a bidirectional LSP. Each unidirectional tunnel can be optionally protected with a protect LSP that activates automatically upon failure conditions.

MPLS Transport Profile Static and Dynamic Multisegment Pseudowires

Multiprotocol Label Switching Transport Profile (MPLS-TP) supports the following combinations of static and dynamic multisegment pseudowires:

- Dynamic-static
- Static-dynamic
- Static-static

MPLS-TP OAM Status for Static and Dynamic Multisegment Pseudowires

With static pseudowires, status notifications can be provided by BFD over VCCV or by the static pseudowire OAM protocol. However, BFD over VCCV sends only attachment circuit status code notifications. Hop-by-hop notifications of other pseudowire status codes are not supported. Therefore, the static pseudowire OAM protocol is preferred

MPLS Transport Profile Links and Physical Interfaces

Multiprotocol Label Switching Transport Profile (MPLS-TP) link numbers may be assigned to physical interfaces only. Bundled interfaces and virtual interfaces are not supported for MPLS-TP link numbers.

The MPLS-TP link creates a layer of indirection between the MPLS-TP tunnel and midpoint LSP configuration and the physical interface. The **mplstp link** command is used to associate an MPLS-TP link number with a physical interface and next-hop node. The MPLS-TP out-links can be configured only on the ethernet interfaces, with either the next hop IPv4 address or next hop mac-address specified.

Multiple tunnels and LSPs may then refer to the MPLS-TP link to indicate that they are traversing that interface. You can move the MPLS-TP link from one interface to another without reconfiguring all the MPLS-TP tunnels and LSPs that refer to the link.

Link numbers must be unique on the router or node.

Tunnel Midpoints

Tunnel LSPs, whether endpoint or midpoint, use the same identifying information. However, it is entered differently.

- At the midpoint, all information for the LSP is specified with the **mpls tp lsp** command for configuring forward and reverse information for forwarding.
- At the midpoint, determining which end is source and which is destination is arbitrary. That is, if you are configuring a tunnel between your device and a coworker's device, then your device is the source. However, your coworker considers his or her device to be the source. At the midpoint, either device could be considered the source. At the midpoint, the forward direction is from source to destination, and the reverse direction is from destination to source.
- At the endpoint, the local information (source) either comes from the global device ID and global ID, or from the locally configured information using the **tp source** command.
- At the endpoint, the remote information (destination) is configured using the **tp destination** command after you enter the **interface tunnel-tp** *number* command. The **tp destination** command includes the

destination node ID, and optionally the global ID and the destination tunnel number. If you do not specify the destination tunnel number, the source tunnel number is used.

- At the endpoint, the LSP number is configured in working-lsp or protect-lsp submode. The default is 0 for the working LSP and 1 for the protect LSP.
- When configuring LSPs at midpoint devices, ensure that the configuration does not deflect traffic back to the originating node.

MPLS-TP Linear Protection with PSC Support

MPLS-TP Linear Protection with PSC Support Overview

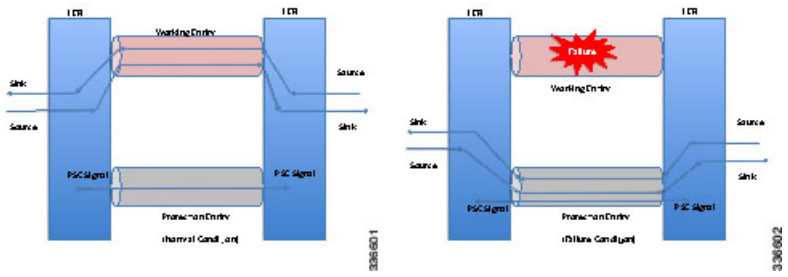
The Multiprotocol Label Switching (MPLS) Transport Profile (TP) enables you to create tunnels that provide the transport network service layer over which IP and MPLS traffic traverse.

Network survivability is the ability of a network to recover traffic deliver following failure, or degradation, of network resources. The MPLS-TP Survivability Framework (RFC-6372) describes the framework for survivability in MPLS-TP networks, focusing on mechanisms for recovering MPLS-TP label switched paths (LSPs)

Linear protection provides rapid and simple protection switching because it can operate between any pair of points within a network. Protection switching is a fully allocated survivability mechanism, meaning that the route and resources of the protection path are reserved for a selected working path or set of working paths. For a point-to-point LSPs, the protected domain is defined as two label edge routers (LERs) and the transport paths that connect them.

Protection switching in a point-to-point domain can be applied to a 1+1, 1:1, or 1:n unidirectional or bidirectional protection architecture. When used for bidirectional switching, the protection architecture must also support a Protection State Coordination (PSC) protocol. This protocol is used to help coordinate both ends of the protected domain in selecting the proper traffic flow. For example, if either endpoint detects a failure on the working transport entity, the endpoint sends a PSC message to inform the peer endpoint of the state condition. The PSC protocol decides what local action, if any, should be taken.

The following figure shows the MPLS-TP linear protection model used and the associated PSC signaling channel for state coordination.



In 1:1 bidirectional protection switching, for each direction, the source endpoint sends traffic on either a working transport entity or a protected transport entity, referred to as a data-path. If the either endpoint detects a failure on the working transport entity, that endpoint switches to send and receive traffic from the protected transport entity. Each endpoint also sends a PSC message to inform the peer endpoint of the state condition. The PSC mechanism is necessary to coordinate the two transport entity endpoints and implement 1:1 bidirectional protection switching even for a unidirectional failure. The switching of the transport path from

working path to protected path can happen because of various failure conditions (such as link down indication (LDI), remote defect indication (RDI), and link failures) or because administrator/operator intervention (such as shutdown, lockout of working/forced switch (FS), and lockout of protection).

Each endpoint LER implements a PSC architecture that consists of multiple functional blocks. They are:

- **Local Trigger Logic:** This receives inputs from bidirectional forwarding detection (BFD), operator commands, fault operation, administration, and maintenance (OAM) and a wait-to-restore (WTR) timer. It runs a priority logic to decide on the highest priority trigger.
- **PSC FSM:** The highest priority trigger event drives the PSC finite state machine (FSM) logic to decide what local action, if any, should be taken. These actions may include triggering path protection at the local endpoint or may simply ignore the event.
- **Remote PSC Signaling:** In addition to receiving events from local trigger logic, the PSC FSM logic also receives and processes PSC signaling messages from the remote LER. Remote messages indicate the status of the transport path from the viewpoint of the far end LER. These messages may drive state changes on the local entity.
- **PSC Message Generator:** Based on the action output from the PSC control logic, this functional block formats the PSC protocol message and transmits it to the remote endpoint of the protected domain. This message may either be the same as the previously transmitted message or change when the PSC control has changed. The messages are transmitted as an initial burst followed by a regular interval.
- **Wait-to-Restore Timer:** The (configurable) WTR timer is used to delay reversion to a normal state when recovering from a failure condition on the working path in revertive mode. The PSC FSM logic starts/stops the WTR timer based on internal conditions/state. When the WTR expires, it generates an event to drive the local trigger logic.
- **Remote Event Expire Timer:** The (configurable) remote-event-expire timer is used to clear the remote event after the timer is expired because of remote inactivity or fault in the protected LSP. When the remote event clear timer expires, it generates a remote event clear notification to the PSC FSM logic.

Interoperability With Proprietary Lockout

An emulated protection (emulated automatic protection switching (APS)) switching ensures synchronization between peer entities. The emulated APS uses link down indication (LDI)message (proprietary) extensions when a lockout command is issued on the working or protected LSP. This lockout command is known as emLockout. A lockout is mutually exclusive between the working and protected LSP. In other words, when the working LSP is locked, the protected LSP cannot be locked (and vice versa).

The emLockout message is sent on the specified channel from the endpoint on the LSP where the lockout command (working/protected) is issued. Once the lockout is cleared locally, a Wait-To-Restore (WTR) timer (configurable) is started and the remote end notified. The local peer continues to remain in lockout until a clear is received from the remote peer and the WTR timer has expired and only then the LSP is considered to be no longer locked out. In certain deployments, you use a large WTR timer to emulate a non-revertive behavior. This causes the protected LSP to continue forwarding traffic even after the lockout has been removed from the working LSP.

The PSC protocol as specified in RFC-6378 is incompatible with the emulated APS implementation in certain conditions. For example, PSC implements a priority scheme whereby a lockout of protection (LoP) is at a higher priority than a forced switch (FS) issued on a working LSP. When an FS is issued and cleared, PSC states that the switching must revert to the working LSP immediately. However, the emulated APS implementation starts a WTR timer and switches after the timer has expired.

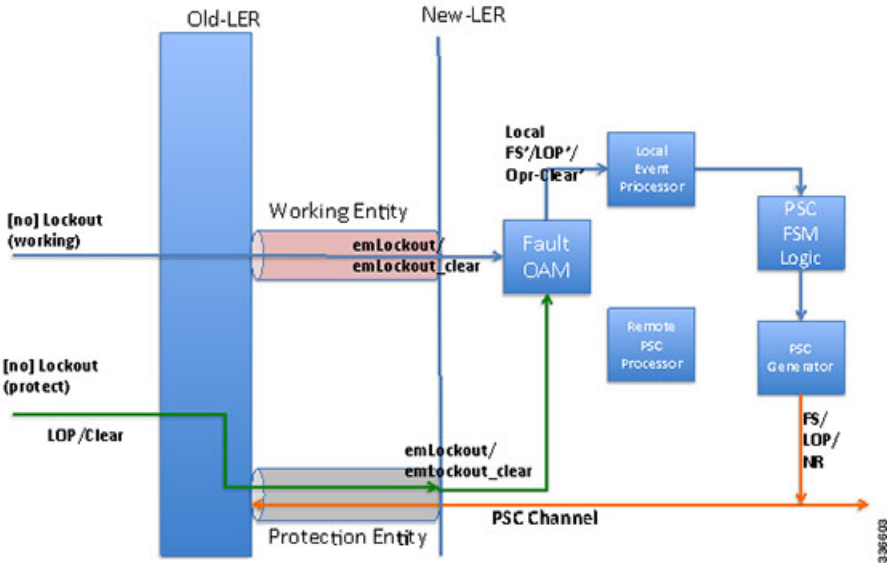
An endpoint implementing the newer PSC version may have to communicate with another endpoint implementing an older version. Because there is no mechanism to exchange the capabilities, the PSC implementation must interoperate with another peer endpoint implementing emulated APS. In this scenario, the new implementation sends both the LDI extension message (referred to as emLockout) as well as a PSC message when the lockout is issued.

Mapping and Priority of emlockout

There are two possible setups for interoperability:

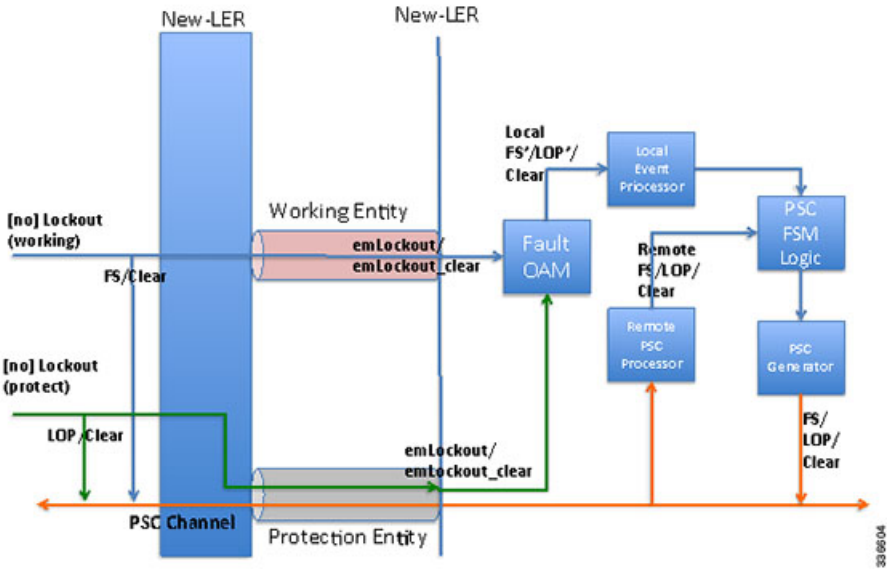
- New-old implementation.
- New-new implementation.

You can understand the mapping and priority when an emLockout is received and processed in the new-old implementation by referring to the following figure.



When the new label edge router (new-LER) receives an emLockout (or emLockout_clear) message, the new-LER maps the message into an internal local FS'/FSc' (local FS-prime/FS-prime-clear) or LoP'/LoPc' (local LoP-prime/Lop-prime-clear) event based on the channel on which it is received. This event is prioritized by the local event processor against any persistent local operator command. The highest priority event drives the PSC FSM logic and any associated path protection logic. A new internal state is defined for FS'/FSc' events. The PSC FSM logic transmits the corresponding PSC message. This message is dropped/ignored by the old-LER.

In the new-new LER implementation shown in the following figure, each endpoint generates two messages when a lockout command is given on a working or protected LSP.



When a lockout (working) command is issued, the new-LER implementation sends an emLockout command on the working LSP and PSC(FS) on the protected LSP. The remote peer receives two commands in either order. A priority scheme for local events is modified slightly beyond what is defined in order to drive the PSC FSM to a consistent state despite the order in which the two messages are received.

In the new implementation, it is possible to override the lockout of the working LSP with the lockout of the protected LSP according to the priority scheme. This is not allowed in the existing implementation. Consider the following steps between old (O) and new (N) node setup:

Time T1: Lockout (on the working LSP) is issued on O and N. Data is switched from the working to the protected LSP.

Time T2: Lockout (on the protected LSP) is issued on O and N. The command is rejected at O (existing behavior) and accepted at N (new behavior). Data in O->N continues on the protected LSP. Data in N->O switches to the working LSP.

You must issue a clear lockout (on the working LSP) and re-issue a lockout (on the protected LSP) on the old node to restore consistency.

WTR Synchronization

When a lockout on the working label switched path (LSP) is issued and subsequently cleared, a WTR timer (default: 10 sec, configurable) is started. When the timer expires, the data path is switched from protected to working LSP.

The PSC protocol indicates that the switch should happen immediately when a lockout (FS) is cleared.

When a new node is connected to the old node, for a period of time equal to the WTR timer value, the data path may be out-of-sync when a lockout is cleared on the working LSP. You should configure a low WTR value in order to minimize this condition.

Another issue is synchronization of the WTR value during stateful switchover (SSO). Currently, the WTR residual value is not checkpointed between the active and standby. As a result, after SSO, the new active restarts the WTR with the configured value if the protected LSP is active and the working LSP is up. As part of the PSC protocol implementation, the residual WTR is checkpointed on the standby. When the standby becomes active, the WTR is started with the residual value.

Priority of Inputs

The event priority scheme for locally generated events is as follows in high to low order:

Local Events:

1. Opr-Clear (Operator Clear)
2. LoP (Lockout of Protection)
3. LoP'/LoP'-Clear
4. FS (Forced Switch)
5. FS'/FS'-Clear
6. MS (Manual-Switch)

The emLockout received on the working LSP is mapped to the local-FS'. The emLockout received on the protected LSP is mapped to the local-LoP'. The emLockout-clear received is mapped to the corresponding clear events.

The priority definition for Signal Fail (SF), Signal Degrade (SD), Manual Switch (MS), WTR, Do Not Revert (DNR), and No Request (NR) remains unchanged.

PSC Syslogs

The following are the new syslogs that are introduced as part of the Linear Protection with PSC Support feature:

SYSLOG NAME	DESCRIPTION	RAW FORMAT
MPLS_TP_TUNNEL_PSC_PREEMPTION	Handle MPLS TP tunnel PSC event preemption syslog.	%MPLS-TP-5-PSCPREEMPTION: Tunnel-tp10, PSC Event: LOP:R preempted PSC Event: FS:L
MPLS_TP_TUNNEL_PSC_TYPE_MISMATCH	Handle MPLS TP tunnel type mismatch	%MPLS-PSC-5-TYPE-MISMATCH: Tunnel-tp10, type mismatch local-type: 1:1,

How to Configure MPLS Transport Profile

Configuring the MPLS Label Range

You must specify a static range of Multiprotocol Label Switching (MPLS) labels using the **mpls label range** command with the **static** keyword.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label range** *minimum-value maximum-value* **static** *minimum-static-value maximum-static-value*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls label range** *minimum-value maximum-value* **static** *minimum-static-value maximum-static-value* **Example:** `Device(config)# mpls label range 1001 1003 static 10000 25000`	Specifies a static range of MPLS labels.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring the Router ID and Global ID

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls tp**
4. **router-id** *node-id*
5. **global-id** *num*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**mpls tp** **Example:** `Device(config)# mpls tp`	Enters MPLS-TP configuration mode, from which you can configure MPLS-TP parameters for the device.
Step 4	**router-id** *node-id* **Example:** `Device(config-mpls-tp)# router-id 10.10.10.10`	Specifies the default MPLS-TP router ID, which is used as the default source node ID for all MPLS-TP tunnels configured on the device.
Step 5	**global-id** *num* **Example:** `Device(config-mpls-tp)# global-id 1`	(Optional) Specifies the default global ID used for all endpoints and midpoints. • This command makes the router ID globally unique in a multiprovider tunnel. Otherwise, the router ID is only locally meaningful. • The global ID is an autonomous system number, which is a controlled number space by which providers can identify each other. • The router ID and global ID are also included in fault messages sent by devices from the tunnel midpoints to help isolate the location of faults.
Step 6	**end** **Example:** `Device(config-mpls-tp)# end`	Exits MPLS-TP configuration mode and returns to privileged EXEC mode.

Configuring Bidirectional Forwarding Detection Templates

The **bfd-template** command allows you to create a BFD template and enter BFD configuration mode. The template can be used to specify a set of BFD interval values. You invoke the template as part of the MPLS-TP tunnel. On platforms that support the BFD Hardware Offload feature and that can provide a 60-ms cutover for MPLS-TP tunnels, it is recommended to use the higher resolution timers in the BFD template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bfd-template single-hop** *template-name*
4. **interval** [**microseconds**] {**both** *time* | **min-tx** *time* **min-rx** *time*} [**multiplier** *multiplier-value*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**bfd-template single-hop** *template-name* **Example:** `Device(config)# bfd-template single-hop mpls-bfd-1`	Creates a BFD template and enter BFD configuration mode.
Step 4	**interval** [**microseconds**] {**both** *time* | **min-tx** *time* **min-rx** *time*} [**multiplier** *multiplier-value*] **Example:** `Device(config-bfd)# interval min-tx 99 min-rx 99 multiplier 3`	Specifies a set of BFD interval values.
Step 5	**end** **Example:** `Device(config-bfd)# exit`	Exits BFD configuration mode and returns to privileged EXEC mode.

Configuring Pseudowire OAM Attributes

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-static-oam class** *class-name*
4. **timeout refresh send** *seconds*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-static-oam class** *class-name* **Example:** `Device(config)# pseudowire-static-oam class oam-class1`	Creates a pseudowire OAM class and enters pseudowire OAM class configuration mode.
Step 4	**timeout refresh send** *seconds* **Example:** `Device(config-st-pw-oam-class)# timeout refresh send 20`	Specifies the OAM timeout refresh interval.
Step 5	**exit** **Example:** `Device(config-st-pw-oam-class)# exit`	Exits pseudowire OAM configuration mode and returns to privileged EXEC mode.

Configuring the Pseudowire Class

When you create a pseudowire class, you specify the parameters of the pseudowire, such as the use of the control word, preferred path and OAM class template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *class-name*
4. **encapsulation mpls**
5. **control-word**
6. **mpls label protocol** [ldp | none]
7. **preferred-path** {**interface tunnel** *tunnel-number* | **peer** {*ip-address* | *host-name*}} [**disable-fallback**]
8. **status protocol notification static** *class-name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *class-name* **Example:** `Device(config)# pseudowire-class mpls-tp-class1`	Creates a pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the encapsulation type.
Step 5	**control-word** **Example:** `Device(config-pw-class)# control-word`	Enables the use of the control word.
Step 6	**mpls label protocol** [ldp | none] **Example:** `Device(config-pw-class)# protocol none`	Specifies the type of protocol.
Step 7	**preferred-path** {**interface tunnel** *tunnel-number* | **peer** {*ip-address* | *host-name*}} [**disable-fallback**] **Example:** `Device(config-pw-class)# preferred-path interface tunnel-tp2`	Specifies the tunnel to use as the preferred path.
Step 8	**status protocol notification static** *class-name* **Example:** `Device(config-pw-class)# status protocol notification static oam-class1`	Specifies the OAM class to use.
Step 9	**end** **Example:** `Device(config-pw-class)# end`	Exits pseudowire class configuration mode and returns to privileged EXEC mode.

Configuring the Pseudowire

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface***interface-id*
4. **service instance** *number* **ethernet** [*name*]
5. **mpls label** *local-pseudowire-label remote-pseudowire-label*
6. **mpls control-word**
7. **backup delay** {*enable-delay-period* | **never**} {*disable-delay-period* | **never**}
8. **backup peer** *peer-router-ip-addr vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface***interface-id* **Example:** `Router(config)# interface gigabitethernet 0/0/4`	Specifies the port on which to create the pseudowire and enters interface configuration mode. Valid interfaces are physical Ethernet ports.
Step 4	**service instance** *number* **ethernet** [*name*] **Example:** `Router(config-if)# service instance 2 ethernet`	Configure an EFP (service instance) and enter service instance configuration) mode. • *number*—Indicates EFP identifier. Valid values are from 1 to 400 • (Optional) **ethernet** *name*—Name of a previously configured EVC. You do not need to use an EVC name in a service instance. **Note** You can use service instance settings such as encapsulation, dot1q, and rewrite to configure tagging properties for a specific traffic flow within a given pseudowire session. For more information, see Ethernet Virtual Connections on the Cisco ASR 903 Router.

	Command or Action	Purpose
Step 5	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-if-xconn)# mpls label 1000 1001`	Configures the static pseudowire connection by defining local and remote circuit labels.
Step 6	**mpls control-word** **Example:** `Device(config-if-xconn)# no mpls control-word`	Specifies the control word.
Step 7	**backup delay** {*enable-delay-period* | **never**} {*disable-delay-period* | **never**} **Example:** `Device(config-if-xconn)# backup delay 0 never`	Specifies how long a backup pseudowire virtual circuit (VC) should wait before resuming operation after the primary pseudowire VC goes down.
Step 8	**backup peer** *peer-router-ip-addr vcid* [**pw-class** *pw-class-name*] [**priority** *value*] **Example:** `Device(config-if-xconn)# backup peer 10.0.0.2 50`	Specifies a redundant peer for a pseudowire virtual circuit (VC).
Step 9	**end** **Example:** `Device(config)# end`	Exits xconn interface connection mode and returns to privileged EXEC mode.

Configuring the MPLS-TP Tunnel

On the endpoint devices, create an MPLS TP tunnel and configure its parameters. See the interface tunnel-tp command for information on the parameters.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel-tp** *number*
4. **description** *tunnel-description*
5. **tp tunnel-name** *name*
6. **tp source** *node-id* [*global-id num*]
7. **tp destination** *node-id* [**tunnel-tp** *num*[**global-id** *num*]]
8. **bfd** *bfd-template*
9. **working-lsp**
10. **in-label** *num*
11. **out-label** *num* **out-link** *num*

12. **exit**
13. **protect-lsp**
14. **in-label** *num*
15. **out-label** *num* **out-link** *num*
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel-tp** *number* **Example:** `Device(config)# interface tunnel-tp 1`	Enters tunnel interface configuration mode. Tunnel numbers from 0 to 999 are supported.
Step 4	**description** *tunnel-description* **Example:** `Device(config-if)# description headend tunnel`	(Optional) Specifies a tunnel description.
Step 5	**tp tunnel-name** *name* **Example:** `Device(config-if)# tp tunnel-name tunnel 122`	Specifies the name of the MPLS-TP tunnel.
Step 6	**tp source** *node-id* [*global-id num*] **Example:** `Device(config-if)# tp source 10.11.11.11 global-id 10`	(Optional) Specifies the tunnel source and endpoint.
Step 7	**tp destination** *node-id* [**tunnel-tp** *num*[**global-id** *num*]] **Example:** `Device(config-if)# tp destination 10.10.10.10`	Specifies the destination node of the tunnel.
Step 8	**bfd** *bfd-template* **Example:**	Specifies the BFD template.

	Command or Action	Purpose
	`Device(config-if)# bfd mpls-bfd-1`	
Step 9	**working-lsp** **Example:** `Device(config-if)# working-lsp`	Specifies a working LSP, also known as the primary LSP.
Step 10	**in-label** *num* **Example:** `Device(config-if-working)# in-label 20000`	Specifies the in-label number.
Step 11	**out-label** *num* **out-link** *num* **Example:** `Device(config-if-working)# out-label 20000 out-link`	Specifies the out-label number and out-link.
Step 12	**exit** **Example:** `Device(config-if-working)# exit`	Exits working LSP interface configuration mode and returns to interface configuration mode.
Step 13	**protect-lsp** **Example:** `Device(config-if)# protect-lsp`	Specifies a backup for a working LSP.
Step 14	**in-label** *num* **Example:** `Device(config-if-protect)# in-label 20000`	Specifies the in label.
Step 15	**out-label** *num* **out-link** *num* **Example:** `Device(config-if-protect)# out-label 113 out-link`	Specifies the out label and out link.
Step 16	**end** **Example:** `Device(config-if-protect)# end`	Exits the interface configuration mode and returns to privileged EXEC mode.

Configuring MPLS-TP LSPs at Midpoints

Note When configuring LSPs at midpoint devices, ensure that the configuration does not deflect traffic back to the originating node.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls tp lsp source** *node-id* [**global-id** *num*] **tunnel-tp** *num* **lsp** {*lsp-num* | **protect** | **working**} **destination** *node-id* [**global-id** *num*] **tunnel-tp** *num*
4. **forward-lsp**
5. **in-label** *num* **out-label** *num* **out-link** *num*
6. **exit**
7. **reverse-lsp**
8. **in-label** *num* **out-label** *num* **out-link** *num*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls tp lsp source** *node-id* [**global-id** *num*] **tunnel-tp** *num* **lsp** {*lsp-num* | **protect** | **working**} **destination** *node-id* [**global-id** *num*] **tunnel-tp** *num* **Example:** `Device(config)# mpls tp lsp source 10.10.10.10 global-id 10 tunnel-tp 1 lsp protect destination 10.11.11.11 global-id 10 tunnel-tp 1`	Enables MPLS-TP midpoint connectivity and enters MPLS TP LSP configuration mode.
Step 4	**forward-lsp** **Example:** `Device(config-mpls-tp-lsp)# forward-lsp`	Enters MPLS-TP LSP forward LSP configuration mode.

	Command or Action	Purpose
Step 5	**in-label** *num* **out-label** *num* **out-link** *num* **Example:** `Device(config-mpls-tp-lsp-forw)# in-label 2000 out-label 2100 out-link 41`	Specifies the in label, out label, and out link numbers.
Step 6	**exit** **Example:** `Device(config-mpls-tp-lsp-forw)# exit`	Exits MPLS-TP LSP forward LSP configuration mode.
Step 7	**reverse-lsp** **Example:** `Device(config-mpls-tp-lsp)# reverse-lsp`	Enters MPLS-TP LSP reverse LSP configuration mode.
Step 8	**in-label** *num* **out-label** *num* **out-link** *num* **Example:** `Device(config-mpls-tp-lsp-rev)# in-label 22000 out-label 20000 out-link 44`	Specifies the in-label, out-label, and out-link numbers.
Step 9	**end** **Example:** `Device(config-mpls-tp-lsp-rev)# end`	Exits the MPLS TP LSP configuration mode and returns to privileged EXEC mode.

Configuring MPLS-TP Links and Physical Interfaces

MPLS-TP link numbers may be assigned to physical interfaces only. Bundled interfaces and virtual interfaces are not supported for MPLS-TP link numbers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **mpls tp link** *link-num*{**ipv4** *ip-address* **tx-mac** *mac-address*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 1/0`	Specifies the interface and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.10.10.10 255.255.255.0`	Assigns an IP address to the interface.
Step 5	**mpls tp link** *link-num*{**ipv4** *ip-address* **tx-mac** *mac-address*} **Example:** `Device(config-if)# mpls tp link 1 ipv4 10.0.0.2`	Associates an MPLS-TP link number with a physical interface and next-hop node. On point-to-point interfaces or Ethernet interfaces designated as point-to-point using the **medium p2p**command, the next-hop can be implicit, so the **mpls tp link**command just associates a link number to the interface. Multiple tunnels and LSPs can refer to the MPLS-TP link to indicate they are traversing that interface. You can move the MPLS-TP link from one interface to another without reconfiguring all the MPLS-TP tunnels and LSPs that refer to the link. Link numbers must be unique on the device or node.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring MPLS-TP Linear Protection with PSC Support

The **psc** command allows you to configure MPLS-TP linear protection with PSC support. PSC is disabled by default. However, it can be enabled by issuing the **psc** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls tp**

4. **psc**
5. **psc fast refresh interval** *time-in-msec*
6. **psc slow refresh interval** *time-in-msec*
7. **psc remote refresh interval** *time-in-sec* **message-count** *num*
8. **exit**
9. **interface tunnel-tp** *number*
10. **psc**
11. **emulated-lockout**
12. **working-lsp**
13. **manual-switch**
14. **exit**
15. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls tp** **Example:** `Device(config)# mpls tp`	Enters Multiprotocol Label Switching (MPLS) Transport Profile (TP) global mode.
Step 4	**psc** **Example:** `Device(config-mpls-tp)# psc`	Enables the PSC Protocol.
Step 5	**psc fast refresh interval** *time-in-msec* **Example:** `Device(config-mpls-tp)# psc fast refresh interval 2000`	Configures the fast refresh interval for PSC messages. • The default is 1000 ms with a jitter of 50 percent. The range is from 1000 ms to 5000 sec.
Step 6	**psc slow refresh interval** *time-in-msec* **Example:** `Device(config-mpls-tp)# psc slow refresh interval 10`	Configures the slow refresh interval for PSC messages. • The default is 5 sec. The range is from 5 secs to 86400 secs (24 hours).

	Command or Action	Purpose
Step 7	**psc remote refresh interval** *time-in-sec* **message-count** *num* **Example:** `Device(config-mpls-tp)# psc remote refresh interval 20 message-count 15`	Configures the remote-event expiration timer. • By default, this timer is disabled. The remote refresh interval range is from 5 to 86400 sec (24 hours). The message count is from 5 to 1000. If you do not specify the message count value, it is set to 5, which is the default.
Step 8	**exit** **Example:** `Device(config-mpls-tp)# exit`	Exits MPLS TP global mode.
Step 9	**interface tunnel-tp** *number* **Example:** `Device(config)# interface tunnel-tp 1`	Creates an MPLS-TP tunnel called *number* and enters TP interface tunnel mode.
Step 10	**psc** **Example:** `Device(config-if)# psc`	Enables PSC. By default, PSC is disabled.
Step 11	**emulated-lockout** **Example:** `Device(config-if)# emulated-lockout`	Enables the sending of emLockout on working/protected transport entities if the **lockout** command is issued on each working/protected transport entity respectively. By default, the sending of emLockout is disabled.
Step 12	**working-lsp** **Example:** `Device(config-if)# working-lsp`	Enters working LSP mode on a TP tunnel interface.
Step 13	**manual-switch** **Example:** `Device(config-if-working)# manual-switch`	Issues a local manual switch condition on a working label switched path (LSP). This can be configured only in working LSP mode on a TP tunnel interface.
Step 14	**exit** **Example:** `Device(config-if-working)# exit`	Exits working LSP mode.
Step 15	**exit** **Example:** `Device(config-if)# exit`	Exits TP interface tunnel mode.

Configuring Static-to-Static Multisegment Pseudowires for MPLS-TP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **point-to-point**
4. **bridge-domain***bridge-id*
5. **neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
6. **mpls label** *local-pseudowire-label remote-pseudowire-label*
7. **mpls control-word**
8. **neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
9. **mpls label** *local-pseudowire-label remote-pseudowire-label*
10. **mpls control-word**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *name* **point-to-point** **Example:** `Device(config)# l2 vfi atom point-to-point`	Creates a point-to-point Layer 2 virtual forwarding interface (VFI) and enters VFI configuration mode.
Step 4	**bridge-domain***bridge-id* **Example:** `Device)config)# bridge-domain 400`	Configures the bridge domain service instance. • *bridge-id*—Bridge domain identifier. The valid values are from 1 to 4000.
Step 5	**neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} **Example:** `Device(config-vfi)# neighbor 10.111.111.111 123 pw-class atom`	Sets up an emulated VC. Specify the IP address, the VC ID of the remote device, and the pseudowire class to use for the emulated VC. **Note** Only two **neighbor** commands are allowed for each Layer 2 VFI point-to-point command.

	Command or Action	Purpose
Step 6	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-vfi)# mpls label 10000 25000`	Configures the static pseudowire connection by defining local and remote circuit labels.
Step 7	**mpls control-word** **Example:** `Device(config-vfi)# mpls control-word`	Specifies the control word.
Step 8	**neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} **Example:** `Device(config-vfi)# neighbor 10.10.10.11 123 pw-class atom`	Sets up an emulated VC. Specify the IP address, the VC ID of the remote device, and the pseudowire class to use for the emulated VC.
Step 9	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-vfi)# mpls label 11000 11001`	Configures the static pseudowire connection by defining local and remote circuit labels.
Step 10	**mpls control-word** **Example:** **Example:** `Device(config-vfi)# mpls control-word`	Specifies the control word.
Step 11	**end** **Example:** `Device(config)# end`	Exits VFI configuration mode and returns to privileged EXEC mode.

Configuring Static-to-Dynamic Multisegment Pseudowires for MPLS-TP

When you configure static-to-dynamic pseudowires, you configure the static pseudowire class with the protocol none command, create a dynamic pseudowire class, and then invoke those pseudowire classes with the neighbor commands.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **pseudowire-class** *class-name*
4. **encapsulation mpls**
5. **control-word**
6. **mpls label protocol** [ldp | none]
7. **exit**
8. **pseudowire-class** *class-name*
9. **encapsulation mpls**
10. **exit**
11. **l2 vfi** *name* **point-to-point**
12. **neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
13. **neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
14. **mpls label** *local-pseudowire-label remote-pseudowire-label*
15. **mpls control-word**
16. **local interface** *pseudowire-type*
17. Do one of the following:
    - **tlv** [*type-name*] *type-value length* [**dec** | **hexstr** | **str**] *value*
    - **tlv template** *template-name*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *class-name* **Example:** `Device(config)# pseudowire-class mpls-tp-class1`	Creates a pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the encapsulation type.
Step 5	**control-word** **Example:** `Device(config-pw-class)# control-word`	Enables the use of the control word.

	Command or Action	Purpose
Step 6	**mpls label protocol** [ldp | none] **Example:** `Device(config-pw-class)# protocol none`	Specifies the type of protocol.
Step 7	**exit** **Example:** `Device(config-pw-class)# exit`	Exits pseudowire class configuration mode and returns to global configuration mode.
Step 8	**pseudowire-class** *class-name* **Example:** `Device(config)# pseudowire-class mpls-tp-class1`	Creates a pseudowire class and enters pseudowire class configuration mode.
Step 9	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the encapsulation type.
Step 10	**exit** **Example:** `Device(config-pw-class)# exit`	Exits pseudowire class configuration mode and returns to global configuration mode.
Step 11	**l2 vfi** *name* **point-to-point** **Example:** `Device(config)# l2 vfi atom point-to-point`	Creates a point-to-point Layer 2 virtual forwarding interface (VFI) and enters VFI configuration mode.
Step 12	**neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} **Example:** `Device(config-vfi)# neighbor 10.111.111.111 123 pw-class atom`	Sets up an emulated VC and enters VFI neighbor configuration mode. **Note** Note: Only two **neighbor** commands are allowed for each **l2 vfi point-to-point** command.
Step 13	**neighbor** *ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} **Example:** `Device(config-vfi-neighbor)# neighbor 10.111.111.111 123 pw-class atom`	Sets up an emulated VC. **Note** Only two **neighbor** commands are allowed for each **l2 vfi point-to-point** command.
Step 14	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:**	Configures the static pseudowire connection by defining local and remote circuit labels.

	Command or Action	Purpose
	`Device(config-vfi-neighbor)# mpls label 10000 25000`	
Step 15	**mpls control-word** **Example:** `Device(config-vfi-neighbor)# mpls control-word`	Specifies the control word.
Step 16	**local interface** *pseudowire-type* **Example:** `Device(config-vfi-neighbor)# local interface 4`	Specifies the pseudowire type.
Step 17	Do one of the following: • **tlv** [*type-name*] *type-value length* [**dec** | **hexstr** | **str**] *value* • **tlv template** *template-name* **Example:** `Device(config-vfi-neighbor)# tlv statictemp 2 4 hexstr 1`	Specifies the TLV parameters or invokes a previously configured TLV template.
Step 18	**end** **Example:** `Device(config-vfi-neighbor)# end`	Ends the session.

Configuring a Template with Pseudowire Type-Length-Value Parameters

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **tlv** [*type-name*] *type-value length* [**dec** | **hexstr** | **str**] *value*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**tlv** [*type-name*] *type-value length* [**dec** | **hexstr** | **str**] *value* **Example:** `Device(config-pw-tlv-template)# tlv statictemp 2 4 hexstr 1`	Specifies the TLV parameters.
Step 4	**end** **Example:** `Device(config-pw-tlv-template)# end`	Exits pseudowire TLV template configuration mode and returns to privileged EXEC mode.

Verifying the MPLS-TP Configuration

Use the following commands to verify and help troubleshoot your MPLS-TP configuration:

- **debug mpls tp**—Enables the logging of MPLS-TP error messages.
- **logging (MPLS-TP)**—Displays configuration or state change logging messages.
- **show bfd neighbors mpls-tp**—Displays the BFD state, which must be up in order for the endpoint LSPs to be up.
- **show mpls l2transport static-oam l2transport static-oam**—Displays MPLS-TP messages related to pseudowires.
- **show mpls tp tunnel-tp** *number* **detail**—Displays the number and details of the tunnels that are not functioning.
- **show mpls tp tunnel-tp lsps**—Displays the status of the LSPs, and helps you ensure that both LSPs are up and working from a tunnel endpoint.
- **traceroute mpls tp** and **ping mpls tp**—Helps you identify connectivity issues along the MPLS-TP tunnel path.

Configuration Examples for MPLS Transport Profile

Example: Configuring MPLS-TP Linear Protection with PSC Support

The following example enters MPLS TP global mode and enables the PSC Protocol.

```
Device> enable
Device# configure terminal
```

```
Device(config)# mpls tp
Device(config-mpls-tp)# psc
```

The following example configures the fast refresh interval for PSC messages. The interval value is 2000 seconds.

```
Device(config-mpls-tp)# psc fast refresh interval 2000
```

The following example configures the slow refresh interval for PSC messages. The interval value is 10 seconds.

```
Device(config-mpls-tp)# psc slow refresh interval 10
```

The following example configures the remote event expiration timer with a refresh interval value of 20 seconds with a message count of 15.

```
Device(config-mpls-tp)# psc remote refresh interval 20 message-count 15
```

The following example exits MPLS TP global mode, creates a TP interface tunnel, and enables PSC.

```
Device(config-mpls-tp)# exit
Decice(config) interface tunnel-tp 1
Device(config-if)# psc
```

The following example enables the sending of emLockout on working/protected transport entities, enters working LSP mode on a TP tunnel interface, and issues a local manual switch condition on a working LSP.

```
Device(config-if)# emulated-lockout
Device(config-if)# working-lsp
Device(config-if-working)# manual-switch
```

Example: Verifying MPLS-TP Linear Protection with PSC Support

The following example displays a summary of the MPLS-TP settings.

```
Device# show mpls tp summary
```

The following example provides information about the MPLS-TP link number database.

```
Device# show mpls tp link-numbers
```

Example: Troubleshooting MPLS-TP Linear Protection with PSC Support

The following example enables debugging for all PSC packets that are sent and received.

```
Device# debug mpls tp psc packet
```

The following example enables debugging for all kinds of PSC events.

```
Device# debug mpls tp psc event
```

The following example clears the counters for PSC signaling messages based on the tunnel number.

```
Device# clear mpls tp 1 psc counter
```

The following example clears the remote event for PSC based on the tunnel number.

```
Device# clear mpls tp tunnel-tp 1 psc remote-event
```