GMPLS UNI

The Generalized Multiprotocol Label Switching (GMPLS) User Network Interface (UNI) creates a circuit connection between two clients (UNI-C) of an optical network. This connection is achieved by signaling exchanges between UNI Client (UNI-C) and UNI Network (UNI-N) nodes. The UNI-C nodes are router nodes and UNI-N nodes are optical nodes.

- Prerequisites for Implementing GMPLS UNI, on page 1
- Restrictions for Implementing GMPLS UNI, on page 1
- Information About Implementing GMPLS UNI , on page 2
- nLight Enhancements, on page 5
- How to Implement GMPLS UNI, on page 9
- Configuration Examples for GMPLS UNI, on page 41
- Additional References, on page 43

Prerequisites for Implementing GMPLS UNI

The following prerequisites are required to implement GMPLS UNI:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- Router that runs Cisco IOS XR software.
- Installation of the Cisco IOS XR software mini-image on the router.
- Installation of the Cisco IOS XR MPLS software package on the router.

Restrictions for Implementing GMPLS UNI

- The total number of configured GMPLS UNI controllers should not exceed the platform scale limit of 500 GMPLS interfaces.
- Each UNI-N (ingress or egress) should be routable from its adjacent UNI-C. The UNI-C nodes need to be routable from the UNI-N nodes too.
- GMPLS UNI is supported only over DWDM controllers and so, over POS and GigabitEthernet interfaces.

- GMPLS UNI is supported only with these Cisco ASR 9000 Enhanced Ethernet Line Cards:
  - A9K-MOD80-SE : 80G Modular Line Card, Service Edge Optimized
  - A9K-MOD80-TR : 80G Modular Line Card, Packet Transport Optimized
  - A9K-36X10GE-SE - Cisco ASR 9000 36-Port 10GE Service Edge Optimized Line Card
  - A9K-36X10GE-TR - Cisco ASR 9000 36-Port 10GE Packet Transport Optimized Line Card
  - A9K-24X10GE-SE - Cisco ASR 9000 24-Port 10GE Service Edge Optimized Line Card
  - A9K-24X10GE-TR - Cisco ASR 9000 24-Port 10GE Packet Transport Optimized Line Card

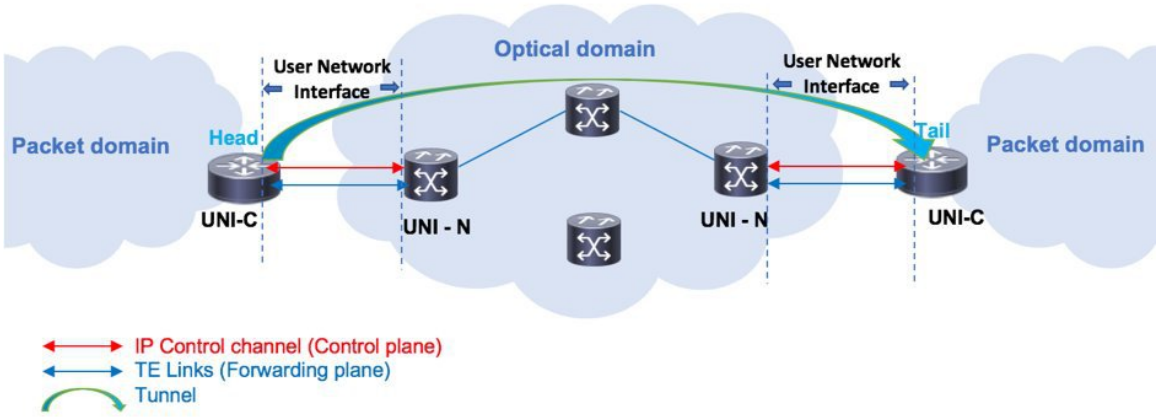
Information About Implementing GMPLS UNI

To implement GMPLS UNI, you should understand these concepts:

GMPLS UNI vs GMPLS NNI

In case of GMPLS NNI, the optical network topology is known and path calculations are performed at the NNI head. In case of GMPLS UNI, the optical network topology is unknown to the UNI-C nodes and path calculations are performed by the UNI-N nodes.

GMPLS UNI Use Case



The UNI components are UNI-N and UNI-C. The tunnel originates on the headend UNI, depicted in the left part of the image. The tunnel terminates on the tailend UNI, depicted in the right part of the image. Enable the following configurations on the headend UNI and tailend UNI:

- Control plane - IP control channel between the UNI-C and UNI-N router IDs. This creates LMP adjacency over the control channel.
- Forwarding plane - TE Link between UNI-C and UNI-N optical interfaces.
- Tunnel configuration from Head UNI-C to a Tail UNI-C optical interface, over the optical network.

For each tunnel, you must enable corresponding tunnel and TE link configurations.

Link Management Protocol (LMP) – LMP manages the control channel across the UNIs, verifies TE link connectivity between the UNI interfaces, and performs fault management.

Dynamic LMP – In release 7.0(1), you can enable the Dynamic LMP function which validates LMP configuration consistency across the headend and tailend UNIs. Consistency check examples:

- You have configured one end of a TE link as an unnumbered interface, and the other end with an IP address.
- You have entered the wrong neighbor interface ID when configuring an unnumbered neighbor interface.

Ensure that you enable the preceding configurations correctly.

GMPLS LSP Signaling

The GMPLS overlay model architecture is used for LSP signaling for GMPLS connections. In GMPLS UNI, UNI-C nodes send a request for a connection to UNI-N node. The connection request does not contain an end-to-end path. This is because, as mentioned previously, UNI-C nodes do not have knowledge of the topology of the optical network and therefore cannot determine the end-to-end path. The UNI-C node signals a connection request without an ERO.

The LSP diversity is signaled on a GMPLS UNI tunnel with a path-option. A path-option is permitted on a GMPLS UNI tunnel with a "no ERO" and an optional "XRO" attribute sets to specify LSP diversity requirements. If multiple LSP exclusions are configured in the attribute-set, they can be added to the path message along with an appropriate LSP connection diversity sub-object.

Release 7.0(1) supports the following LSP encoding and corresponding switching types.

Switching Type	LSP Encoding Type
LSC	Lambda
FSC	EthernetType1, EthernetType2, Fiber
DCSC	EthernetType2

A packet network is switched across a fiber, optic, or data channel network. Enable the LSP encoding and switching types under the GMPLS UNI and LMP configuration modes. Also, enable the Generalized PID (G-PID) under the GMPLS UNI configuration mode. G-PID is an identifier of the type of payload that the LSP carries, and the LSP endpoints (the UNI-C devices) use.

The LSP encoding, switching type, and G-PID are updated to the GMPLS label.

Path Message without an ERO

In GMPLS UNI, UNI-C nodes send a request for a connection to UNI-N node. The connection request does not contain an end-to-end path, because, UNI-C nodes do not have knowledge of the topology of the optical network and therefore cannot determine the end-to-end path. The UNI-C node signals a connection request without an ERO.

When no ERO is present in a received path message, the UNI-N node calculates a route to the destination and includes that route in an ERO, before forwarding the path message. If no route is found, the UNI-N returns a path error message with an error code and subcode of *24,5 - "No route available toward destination"*.

The destination address of a GMPLS LSP can be either the optical router-id of the tail UNI-C node, or the optical address of the ingress interface to the tail UNI-C node. Supplying the router-id allows the UNI-N to route the tunnel to the tail UNI-C node via any attached UNI-N node; supplying the UNI-C's ingress interface address forces the tunnel's path to traverse the UNI-N node attached to that interface.

Note The optical router-ids and interface addresses may or may not be the same as the packet ones.

XRO Attribute-set

An optional XRO attribute-set can be specified as part of the path-option to specify LSP diversity requirements. An empty XRO attribute set results in the GMPLS tunnel being signaled with no exclusions, and therefore no XRO.

Note A non-existent XRO attribute-set can be configured in the GMPLS UNI tunnel path-option; in this case no attempt will be made to bring up the GMPLS tunnel until the configuration is complete.

Connection Diversity

Connection diversity is required to ensure that GMPLS tunnels can be established without sharing resources, thus, greatly reducing the probability of simultaneous connection failures. For example, an edge-node wishes to establish multiple LSPs towards the same destination edge-node, and these LSPs need to have few or no resources in common.

Connection diversity supports the establishment of a GMPLS LSP which is diverse from the path taken by an existing LSP. An XRO is added to the tunnel's path message with appropriate LSP diversity sub-objects or exclusions. A maximum of 20 connection diversity exclusions per XRO is supported.

GMPLS RSVP VRF Signaling

The Cisco IOS XR software supports a single non-default VRF for the GMPLS RSVP signaling. This allows GMPLS signaling to work even when the only available communication between the UNI-C and UNI-N nodes is through a VRF. This non-default VRF is supported only for GMPLS signaling; whereas the MPLS-TE signaling continues to support only the default VRF.

DWDM Transponder Integration

A GMPLS UNI based solution preserves all the advantages of the integration of the DWDM transponder into the router blade. These advantages include:

- improved CAPEX and OPEX models
- component, space and power savings
- improved IP availability through pro-active protection.

nLight Enhancements

These topics describe the enhancements made to nLight (also known as GMPLS UNI):

Explicit Route Object

Explicit Route Objects (EROs) limit LSP routing to a specified list of LSRs. Formerly, the UNI Client (UNI-C) node signaled a connection request, without an ERO, to the UNI Network (UNI-N) node. In this IOS XR Software release, the UNI-C node provides support for path message with ERO for GMPLS tunnels. This includes the capability to specify either a strict or a loose ERO to a path option to be included in the path message for processing by the ingress UNI-N.

An ERO in constructed using the strict and loose hops, specified in the explicit path, by the path option.

When a loose hop is configured, it identifies one or more transit LSRs which suggests the preferred path for the LSP. If a suggested path fails, another LSR is tried.

When a strict hop is configured, it identifies an exact path through which the LSP must be routed. Strict hop EROs specify the exact sequence of LSRs in the LSP.

As a result of these operations, a LSP is established from the sender to the destination of the session, following the explicitly routed path specified in the ERO.

Note

- *lockdown* and *verbatim* are mandatory in ERO path option.
- A path option may still be configured to use no ERO.
- In no ERO, *lockdown* is mandatory.

Wavelength Specification

The wavelength (also called label) specification enhancement enables the network planning tool to determine the wavelength, and specify the same at the UNI-C. The UNI-N then accepts the label provided by the UNI-C, or rejects the path entirely. Previously, the wavelength to be used for the GMPLS UNI tunnel was determined by the UNI-N, taking into account the headend UNI-C's capabilities.

The wavelength to be used is added to the path option configuration. This optional configuration allows a fixed wavelength to be specified for the path option.

When signaling using a path option with the specified wavelength takes place, the following changes happen because of the wavelength specification enhancement:

- The configured wavelength is validated against the controller’s capabilities; signaling fails if the wavelength cannot be used by the controller.
- The upstream label is set to the specified wavelength.
- The label-set in the Path message, instead of containing one label for each supported wavelength, contains only the specified wavelength.

- A path-error message with error code 25 and subcode 6 no longer receives special handling. If a suggested label is supplied, it is ignored.

Note A suggested label received in response to signaling with a path option that specifies a different label, is not stored for future use. Other path options, in general, have different constraints and therefore require path calculation to be redone.

Multiple Path Options

Multiple path options are permitted per GMPLS UNI tunnel. The index given to each path option indicates its relative preference level, with lower indices being preferred. This is similar to the existing multiple path option functionality available for packet TE. This allows the provision of multiple path options with, for example, progressively free constraints.

The path-option index is no longer fixed to ten and is now set by the user and distinguishes path options in the same manner as for packet tunnels. In all situations where a tunnel is being brought up or reoptimized, the path-option with the lowest index is tried first; if no LSP can be established with this path option, then subsequent path options are tried in ascending order. This also applies to recovery from failures, unless any recovery path option is specified.

Reoptimization

Reoptimization differs from restoration though the mechanisms involved are similar. Reoptimization occurs without the original connection having failed.

Unlike packet tunnels, reoptimization in GMPLS tunnels is not supposed to be loss free.

Manual Reoptimization

Manual reoptimazation of a single GMPLS UNI tunnel can be triggered from the UNI-C node (headend). Use the **mpls traffic-eng optical-uni reoptimize tunnel-id** command to trigger manual reoptimization of a GMPLS UNI tunnel.

The manual trigger for reoptimization causes the currently established LSP to be torn down and signals a new LSP using the normal bring-up process (though the new LSP is same as the current one).

It is not possible to trigger reoptimization for multiple GMPLS UNI tunnels or at the tailend of a tunnel.

SRLG Discovery

Note SRLG (Shared Risk Link Group) discovery, SRLG collection and SRLG recording represent the same function.

The head and tail UNI-C routers have no direct knowledge of the path taken through the optical network by a GMPLS UNI tunnel, or of the properties of that path. All information about the path of a particular GMPLS UNI connection must therefore be explicitly requested and learned during the signaling process.

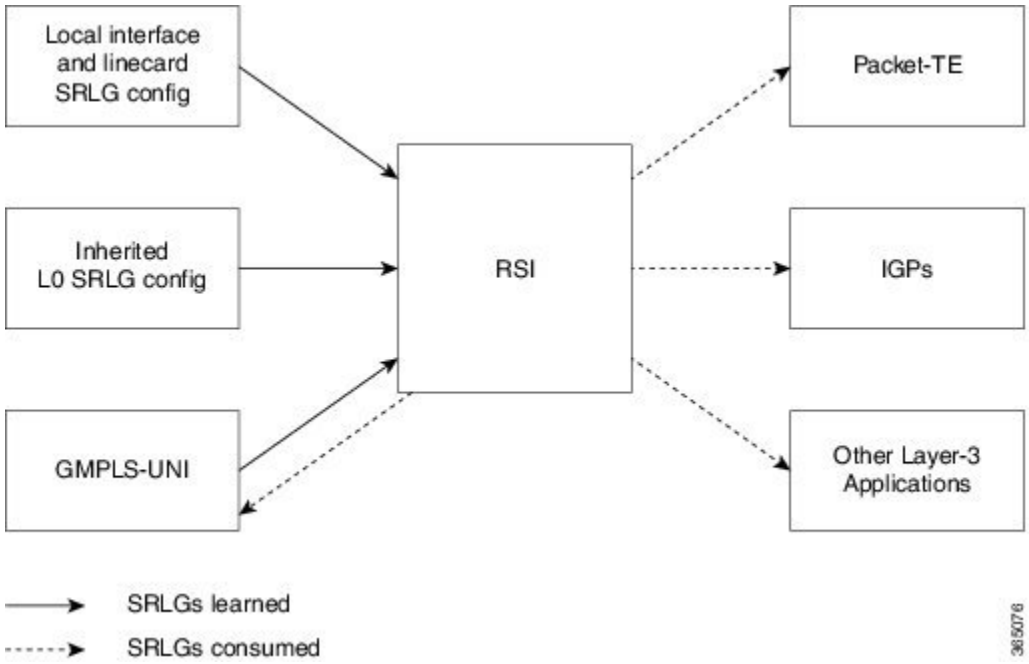
A key property of a GMPLS UNI connection is the set of SRLGs used by the optical links along the connection. It is necessary for the UNI-C routers to learn the set of SRLGs associated with a connection, so that this information can be used, both by GMPLS UNI in the specification of diversity requirements for other connections and by Layer-3 applications for effecting routing and protection decision making.

The learning of SRLGs during GMPLS UNI LSP signaling is done by requesting SRLG collection when LSP signaling is initiated, and by the addition of SRLG RRO sub-objects to the Path and Resv messages during signaling as described in IETF draft *SRLG-collect*. Path message learns egress interfaces from head to tail and Resv message learns egress interfaces from tail to head

Provision of Discovered SRLGs to RSI

Once the SRLGs used by a GMPLS UNI connection are collected during signaling as in SRLG discovery, they are made available to the Layer-3 processes. This is done through RSI (Router Space Infrastructure), as illustrated in the following diagram:

Figure 1: SRLG Communication



An API is provided by the RSI component to allow SRLGs discovered during GMPLS UNI signaling to be communicated to RSI, as documented in IETF draft *RSI-SRLG*. RSI combines the SRLG sets learned from GMPLS and configuration for an interface and deliver a single set of SRLGs to applications registered as SRLG clients.

The SRLGs discovered during GMPLS UNI signaling are given to RSI for application to the Layer-3 interface of the DWDM controller associated with the GMPLS UNI tunnel. This may be a POS, GigE or an OTN interface.

SRLG Announce

All SRLGs discovered through GMPLS signaling are announced to RSI once the tunnel is up. These SRLGs are withdrawn from RSI when the tunnel goes down.

SRLG Diversity

Note SRLG diversity and SRLG exclusion represent the same function.

Support is added for signaling SRLG based diversity requirements, based on the XRO SRLG sub-object defined in RFC 4874. The use of SRLGs removes the restrictions of LSP based diversity, as SRLGs are flooded throughout the optical network, and by their very nature, reduce the risk of concurrent failure.

SRLG diversity is configured under the XRO attribute-set.

Head UNI-C Behavior

SRLG diversity is configured at the tunnel head. Individual SRLG exclusions are added to an XRO attribute-set; each is specified as either *best-effort* or mandatory (*strict*). Whenever any exclusion is specified, an XRO object is added to the Path message by the head UNI-C. The XRO contains a SRLG sub-object for each specified SRLG. The SRLG exclusions may coexist in the same XRO with LSP exclusions.

The XRO attribute-set is associated with tunnel path options in the same manner as for LSP exclusions.

If a SRLG with a strict exclusion matches an SRLG configured on the local DWDM controller, the bring-up attempt fails.

The SRLG exclusions requested by the head UNI-C are processed by the ingress UNI-N node during path calculation for the tunnel.

Tail UNI-C Behavior

On receiving a Path message containing an XRO, the tail UNI-C inspects each SRLG sub-object. If a SRLG sub-object, with a strict exclusion, matches an SRLG configured on the local DWDM controller, the Path message is rejected and a path-error is generated with error codes. No action is taken if the SRLG sub-object specifies a *best-effort* exclusion.

Multi-Layer Restoration - Optical

Multi-Layer Restoration-Optical (MLR-O) involves restoration from failures in the optical network that can leverage the same router interfaces at both ends.

Optical restoration involves the repair of a failure by the optical network locally. Although the routers may see loss of light until the failure is repaired, there is no signaling involving the routers, and from the routers perspective the GMPLS UNI LSP remains unchanged.

Optical Restoration: Same Wavelength

When a failure occurs on a physical link within the optical network, the routers identify that the link is down and Layer 3 protection mechanisms, such as FRR, are used to minimize the traffic loss. The optical network re-routes the GMPLS connection to an alternative path. This is done without any involvement of the routers.

Limitation

A significant limitation of optical restoration in this case, is that the wavelength in use for the connection cannot be changed. This is because the wavelength must be the same along the entire path and cannot be

changed without end-to-end signaling. The constraints imposed on the connection during its initial signaling are also unchanged, which may reduce the chance of finding an alternative path.

Optical Restoration: Wavelength Change

Optical restoration may occur with an associated wavelength change, in the case where the optical network finds an alternative path with the same constraints as were originally signaled, but using a different wavelength. Some signaling is required, since the wavelength (and therefore the labels) used by the GMPLS connection are to change.

Consider a failure within the optical network on the path of a GMPLS UNI LSP. The restoration proceeds as in the previous case (same wavelength), but the new path found, uses a different wavelength. The ingress UNI-N then sends a path-error message indicating the new wavelength to be used; this has error code 24 (routing), sub-error 6 (unacceptable label set) and contains a suggested-label sub-object with the new label to be used. The head UNI-C then signals a new LSP with the new wavelength.

Although the wavelength in use may change in this case, the constraints used in signaling the original LSP remain unchanged.

How to Implement GMPLS UNI

A new submode is introduced under the main TE submode to enable GMPLS UNI and to contain GMPLS UNI configuration.

To implement GMPLS UNI, follow these procedures:

Configuring TE for GMPLS UNI

TE configuration specific to packet tunnels does not affect GMPLS UNI tunnels.

To implement TE configuration for GMPLS UNI, follow these procedures:

Enabling GMPLS UNI Submode

Perform this task to enable GMPLS UNI configuration submode and to configure GMPLS UNI tunnels.

Note Removal of the GMPLS UNI submode results in the removal of all configuration within it, including any other parser submode, and the immediate destruction of all GMPLS UNI tunnels.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.

Step 3	**gmpls optical-uni**
	Example:
	`RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`
	`RP/0/RSP0/CPU0:router(config-te-gmpls)#`
		Enters GMPLS UNI configuration submode.

Step 4	**commit**	

Configuring GMPLS UNI Controller

Perform this task to setup a GMPLS tail in MPLS-TE configuration. This task enables GMPLS UNI controller submode to configure controllers for establishing GMPLS UNI tunnels. This is the minimal configuration required at the tunnel tail.

Note Removal of the GMPLS UNI controller submode results in the immediate destruction of any GMPLS tunnel established over the controller referenced.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.

Step 4	**controller dwdm** *interface*
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls)# controller dwdm 0/1/0/1`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)#`
		Enters GMPLS UNI controller submode.

Step 5	**commit**	

Configuring the GMPLS UNI Controller as a Tunnel Head

Perform this task to configure the tunnel properties for a GMPLS UNI controller.

This configuration designates the controller as a tunnel-head, rather than a tunnel tail. After you configure the tunnel properties, the incoming path messages are rejected and any existing tailend tunnel is torn down.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **mtu** *value*
6. **tunnel-properties**
7. **g-pid** *ID*
8. **encoding-type** *type*
9. **tunnel-id** *number*
10. **destination ipv4 unicast** *address*
11. **path-option 10 no-ero lockdown**
12. **exit**
13. **switching-type** *type*
14. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.

Step 4	**controller dwdm** *interface*
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls)# controller dwdm 0/1/0/1`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)#`
		Enters GMPLS UNI controller submode.

Step 5	**mtu** *value* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# mtu 9000`	Enable the maximum traffic limit on the interface.

Step 6	**tunnel-properties**
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`
		Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

Step 7	**g-pid** *ID* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# g-pid 37`	Assigns the Generalized PID (G-PID) which is an identifier of the payload that is carried by the LSP. The LSP endpoints (the UNI-C devices) use the G-PID.
Step 8	**encoding-type** *type* **Example:**	Assigns the LSP encoding type.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# encoding-type lambda`	
Step 9	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 100`	Specifies a tunnel-id for a headend router of a GMPLS tunnel. The tunnel-id is a 16-bit number ranging 0–65535.
Step 10	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 10.10.3.4`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 11	**path-option 10 no-ero lockdown** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 no-ero lockdown`	Specifies the path-option for a headend router of a GMPLS tunnel. **Note** You can specify an XRO attribute-set as part of the path-option.
Step 12	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# exit`	Exits the mode and enters the controller mode.
Step 13	**switching-type** *type* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# switching-type lsc`	Assigns the switching type of the LSP traffic.
Step 14	**commit**	

Configuring Other Tunnel Properties for a GMPLS UNI Tunnel

Perform this task to configure the optional tunnel properties for a GMPLS UNI tunnel. This configuration is optional, and if omitted, the GMPLS tunnel is established with the default property values.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **priority** *setup-priority hold-priority*
7. **record-route**

8. **signalled-name** *name*
9. **logging events lsp-status state**
10. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls)# controller dwdm 0/1/0/1`	Enters GMPLS UNI controller submode.
Step 5	**tunnel-properties** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`	Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.
Step 6	**priority** *setup-priority hold-priority* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# priority 3 2`	Specifies the priority for a GMPLS tunnel. The default priority value is 7 for both setup and hold priorities. **Note** The setup-priority and hold-priority values are numbers ranging from 0 to 7, where 0 represents the highest priority. The hold-priority must be equal or higher (numerically less) than the setup-priority.
Step 7	**record-route** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# record-route`	Enables record-route functionality for a GMPLS tunnel.

	Command or Action	Purpose
Step 8	**signalled-name** *name* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# signalled-name sign1`	Configures signalled-name for a GMPLS tunnel. **Note** If no signalled name is configured, TE will generate a default name in the form of *router-name_tunnel-id_destination-address*, for example, te-ma1_123_10.10.10.10.
Step 9	**logging events lsp-status state** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# logging events lsp-status state`	Configure events to generate system log messages when state changes occur on the GMPLS tunnel. If omitted, no events will result in the generation of system log messages.
Step 10	**commit**	

Configuring LSP Diversity

To configure an XRO attribute-set as part of the path-option for MPLS-TE, and to specify exclusions for an attribute set for LSP diversity, follow these procedures:

Configuring XRO Attribute-set

Perform this task to configure XRO attribute set in the GMPLS UNI tunnel path-option, under MPLS-TE submode.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **path-option 10 no-ero [xro-attribute-set** *name* **] lockdown**
7. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.

	Command or Action	Purpose
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls)# controller dwdm 0/1/0/1`	Enters GMPLS UNI controller submode.
Step 5	**tunnel-properties** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`	Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.
Step 6	**path-option 10 no-ero [xro-attribute-set** *name* **] lockdown** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 no-ero xro-attribute-set A01 lockdown`	Specifies the path-option for a headend router of a GMPLS tunnel.
Step 7	**commit**	

Configuring Connection Diversity

Perform this task to specify exclusions for an attribute set for LSP diversity, under MPLS-TE attribute-set configuration mode.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **attribute-set xro** *name*
4. **exclude** {**best-effort** | **strict**} **lsp source** *source-address* **destination** *destination-address* **tunnel-id** *tunnel-id* **extended-tunnel-id** *extended-tunnel-id* [**lsp-id** *lsp-id*]
5. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**attribute-set xro** *name* **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# attribute-set xro attrset01`	Configures an XRO attribute-set for a GMPLS tunnel.

Step 4	**exclude** {**best-effort** | **strict**} **lsp source** *source-address* **destination** *destination-address* **tunnel-id** *tunnel-id* **extended-tunnel-id** *extended-tunnel-id* [**lsp-id** *lsp-id*]
	Example:
	`RP/0/RSP0/CPU0:router(config-te-attribute-set)# exclude best-effort lsp source 10.10.1.2 destination 10.20.4.4 tunnel-id 17 extended-tunnel-id 10.20.3.3 lsp-id 17`
	`RP/0/RSP0/CPU0:router(config-te-attribute-set)#`
		Specifies exclusions for an attribute set for LSP diversity.
		Note
		A maximum of 20 LSP exclusions per XRO is supported.

Step 5	**commit**	

Configuring LMP for GMPLS UNI

To implement LMP configuration for GMPLS UNI, follow these procedures:

Configuring Optical Router ID

Perform this task to enable GMPLS UNI LMP functionality and to configure LMP unicast router ID.

SUMMARY STEPS

1. **configure**
2. **lmp**
3. **gmpls optical-uni**
4. **router-id ipv4 unicast** *address*
5. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**lmp** **Example:** `RP/0/RSP0/CPU0:router(config)# lmp`	Enters LMP configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-lmp)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**router-id ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni)# router-id ipv4 unicast 10.10.4.4`	Configures the LMP unicast router ID for GMPLS.
Step 5	**commit**	

Configuring an LMP Neighbor

Perform this task to configure an LMP neighbor for a GMPLS UNI tunnel.

SUMMARY STEPS

1. **configure**
2. **lmp**
3. **gmpls optical-uni**
4. **neighbor** *name*
5. **dynamic**
6. **hello** *interval dead-interval*
7. **ipcc routed**
8. **router-id ipv4 unicast** *address*
9. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**lmp** **Example:**	Enters LMP configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config)# lmp`	
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-lmp)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**neighbor** *name* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni)# neighbor nbr1`	Specifies an LMP neighbor for GMPLS and enters the LMP GMPLS UNI neighbor configuration submode.
Step 5	**dynamic** **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-nbr1)# dynamic`	Configures Dynamic LMP function.
Step 6	**hello** *interval dead-interval* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-nbr1)# hello 1000 10000`	Specifies the LMP hello message frequency (in milliseconds) sent between LMP enabled routers. It also specifies the time duration (in milliseconds) after which the device sends an LMP hello expiry message. The LMP hello expiry message duration must be three times more than the LMP hello interval duration. If you do not use the LMP fast keep-alive mechanism, ensure that you set the two interval values to zero.
Step 7	**ipcc routed** **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-nbr-nbr1)# ipcc routed`	Specifies the LMP neighbor IPCC configuration for GMPLS UNI.
Step 8	**router-id ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-nbr-nbr1)# router-id ipv4 unicast 10.10.4.5`	Configures the LMP unicast router ID for GMPLS.
Step 9	**commit**	

Configuring an LMP Controller

Perform this task to configure an LMP link for a GMPLS UNI controller.

SUMMARY STEPS

1. **configure**
2. **lmp**
3. **gmpls optical-uni**
4. **controller dwdm** *controller*
5. **neighbor** *name*
6. **link-id ipv4 unicast** *address*
7. **switching-type** *type*
8. **encoding-type** *type*
9. **neighbor link-id ipv4 unicast** *address*
10. **neighbor interface-id unnumbered** *interface-id*
11. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**lmp** **Example:** `RP/0/RSP0/CPU0:router(config)# lmp`	Enters LMP configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-lmp)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *controller* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni)# controller dwdm 0/4/0/0`	Specifies a controller for GMPLS UNI.
Step 5	**neighbor** *name* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)#`	Specifies an LMP neighbor for GMPLS and enters the LMP GMPLS UNI neighbor configuration submode.

	Command or Action	Purpose
	`neighbor nbr1`	
Step 6	**link-id ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)# link-id ipv4 unicast 10.2.2.4`	Specifies the optical interface address for an LMP link for a GMPLS UNI controller.
Step 7	**switching-type** *type* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)# switching-type lsc`	Specifies type of switching traffic that the LSP carries.
Step 8	**encoding-type** *type* **Example:** `(RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)# encoding-type Lambda`	Specifies the signaling technology that you use for transporting traffic.
Step 9	**neighbor link-id ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)# neighbor link-id ipv4 unicast 10.2.2.5`	Specifies the neighbor's optical address of an LMP link for a GMPLS UNI controller.
Step 10	**neighbor interface-id unnumbered** *interface-id* **Example:** `RP/0/RSP0/CPU0:router(config-lmp-gmpls-uni-cntl)# neighbor interface-id unnumbered 17`	Specifies the neighbor's optical interface ID of an LMP link for a GMPLS UNI controller.
Step 11	**commit**	

Configuring RSVP Optical Refresh Interval and Missed Count

Perform this task to configure optical refresh interval under the RSVP controller submode and to configure the number of missed refresh messages allowed before optical tunnel states are deleted.

SUMMARY STEPS

1. **configure**
2. **rsvp**
3. **controller dwdm** *interface*
4. **signalling refresh out-of-band interval** *interval*

5. **signalling refresh out-of-band missed** *miss-count*
6. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp** **Example:** `RP/0/RSP0/CPU0:router(config)# rsvp`	Enters RSVP configuration mode.
Step 3	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-rsvp)# controller dwdm 0/1/0/1`	Configures a controller for establishing a GMPLS UNI tunnel.
Step 4	**signalling refresh out-of-band interval** *interval* **Example:** `RP/0/RSP0/CPU0:router(config-rsvp-cntl)# signalling refresh out-of-band interval 200`	Configures optical refresh interval. The interval argument is the interval (in seconds) at which refresh messages are sent and expected to be received. The range is 180 to 86400 (a refresh-interval of 1 day).
Step 5	**signalling refresh out-of-band missed** *miss-count* **Example:** `RP/0/RSP0/CPU0:router(config-rsvp-cntl)# signalling refresh out-of-band missed 30`	Configures number of missed refresh messages allowed before optical tunnel states are deleted. The miss-count argument is the number of refresh messages, expected at the configured refresh-interval, which can be missed before optical tunnel states time out. The accepted range is 1 to 48. The default value is 12.
Step 6	**commit**	

nLight Enhancements: Configurations and Verifications

These topics describe the configurations and verifications for the nLight enhancements made:

Configuring an ERO for a GMPLS Tunnel

Perform this task to configure an ERO for a GMPLS tunnel.

SUMMARY STEPS

1. **configure**

2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **tunnel-id** *number*
7. **logging events lsp-status state**
8. **destination ipv4 unicast** *address*
9. **path-option** *number* **explicit name** *name* **lockdown verbatim**
10. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/2/1/1`	Enters GMPLS UNI controller submode.

Step 5	**tunnel-properties**
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`
		Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

	Command or Action	Purpose
Step 6	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 1001`	Specifies a tunnel-ID for a headend router of a GMPLS tunnel. The tunnel-ID is a 16-bit number ranging from 0 to 65535.
Step 7	**logging events lsp-status state** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# logging events lsp-status state`	Configure events to generate system log messages when state changes occur on the GMPLS tunnel. If omitted, no events will result in the generation of system log messages.
Step 8	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 102.3.233.1`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 9	**path-option** *number* **explicit name** *name* **lockdown verbatim** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 explicit name explicit_path_a lockdown verbatim`	Specifies an explicit path for a headend router of a GMPLS tunnel. The path-option range is 1 to 1000. **Note** lockdown and verbatim are mandatory in ERO path option.
Step 10	**commit**	

Verifying an ERO Configuration: Example

The following example shows how to verify an ERO configuration:

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0  Destination: 172.16.0.1
  Signalled-Name: head_ot1001_172.16.0.1
GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN verbatim) type explicit explicit_path_a (Basis for Setup,
path weight 0)
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Fri Jul 17 08:41:21 ---- (3d07h ago)
 ……
  Current LSP Info:
    Instance: 20
    Uptime: 00:00:33 (since Mon Jul 20 ---- 15:45:22)
    Upstream label:
      Optical label:
      Grid               : DWDM
      Channel spacing    : 50 GHz
```

```
    Identifier          : 0
    Channel Number      : 60
  Downstream label:
    Optical label:
    Grid                : DWDM
    Channel spacing     : 50 GHz
    Identifier          : 0
    Channel Number      : 60
  Router-IDs: local      10.0.0.1
              downstream 172.16.0.1
  Soft Preemption: None
  SRLGs: not collected
  Path Info:
    Outgoing:
      Explicit Route:
        Strict, 10.10.10.2
        Strict, 11.11.11.3
        Strict, 12.12.12.3
```

…….

Configuring Wavelength for a Path Option

Perform this task to configure wavelength for a path option for a GMPLS tunnel.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **tunnel-id** *number*
7. **destination ipv4 unicast** *address*
8. **path-option** *number* **explicit name** *name* **signaled-label dwdm wavelength** *dwdm channel number* **lockdown verbatim**
9. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.

	Command or Action	Purpose
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/3/0/0`	Enters GMPLS UNI controller submode.

Step 5	**tunnel-properties**
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`
		Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

Step 6	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 1001`	Specifies a tunnel-ID for a headend router of a GMPLS tunnel. The tunnel-ID is a 16-bit number ranging from 0 to 65535.
Step 7	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 172.16.0.1`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 8	**path-option** *number* **explicit name** *name* **signaled-label dwdm wavelength** *dwdm channel number* **lockdown verbatim** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 explicit name exp_all_loose_hop signaled-label dwdm wavelength 10 lockdown verbatim`	Specifies a wavelength for the path option. The DWDM channel number range is 1 to 89. The DWDM channel number configured is formulated as 61-channel number. So, if we want channel number 42 (in the supported channel list), the configured "DWDM channel number" will be 61 - 42 = 19.
Step 9	**commit**	

Configuring and Verifying Wavelength Configuration: Examples

The following sequence of examples show how to add a wavelength to a path option for a GMPLS tunnel and verify the outgoing label is set accordingly.

This example shows how to configure a GMPLS tunnel with no ERO path option.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 10 no-ero lockdown
   !
  !
 !
```

This example shows how to verify the default values for the outgoing label (UNI-N source, channel number same as Default Channel) and the list of valid wavelengths.

```
RP/0/RP0/CPU0:router#show mpls traffic-eng link-management optical-uni controller dwdm
0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Head
    Tunnel-id: 1001, LSP-id 21, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 10.0.0.1, Remote: 172.16.0.1
    Label source: UNI-N
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 60
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 60
    SRLG discovery: Disabled
    SRLG announcement: None
...
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
```

```
    Controller SRLGs
      None
```

This example shows how to set valid wavelength 10 (61 - 51) for the current path.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
   path-option 10 explicit name explicit_all_loose_multi_hop signaled-label dwdm wavelength
 10 lockdown verbatim
   !
  !
 !
```

This example shows how to verify that the tunnel is up and the specified wavelength is used (label source is UNI-C and outgoing label is 51).

```
RP/0/RP0/CPU0:router#show mpls traffic-eng link-management optical-uni controller dwdm
0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Head
    Tunnel-id: 1001, LSP-id 23, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 10.0.0.1, Remote: 172.16.0.1
    Label source: UNI-C
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    SRLG discovery: Disabled
    SRLG announcement: None
...
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
```

```
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
    Controller SRLGs
      None
```

This example shows how to verify the upstream label on the tunnel tail.

```
RP/0/RP0/CPU0:router#show mpls traffic-eng link-management optical-uni controller dwdm
0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Tail
    Tunnel-id: 1001, LSP-id 23, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 172.16.0.1, Remote: 10.0.0.1
    Label source: UNI-N
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    SRLG discovery: Disabled
    SRLG announcement: None
…..
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
    Controller SRLGs
      None
```

Configuring Multiple Path Options

Perform this task to configure multiple path options for a single tunnel.

Note If a tunnel is up and a lower index path option is configured, the tunnel does not try the lower index path option, unless for some reason the tunnel is flapped or reoptimized.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **tunnel-id** *number*
7. **logging events lsp-status state**
8. **destination ipv4 unicast** *address*
9. **path-option** *number* **explicit name** *name* **lockdown verbatim**
10. **path-option** *number* **explicit name** *name* **lockdown verbatim**
11. **path-option** *number* **no-ero lockdown**
12. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:**	Enters GMPLS UNI controller submode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/3/0/0`	

Step 5	**tunnel-properties**
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`
		Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

Step 6	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 1001`	Specifies a tunnel-ID for a headend router of a GMPLS tunnel. The tunnel-ID is a 16-bit number ranging from 0 to 65535.
Step 7	**logging events lsp-status state** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# logging events lsp-status state`	Configure events to generate system log messages when state changes occur on the GMPLS tunnel. If omitted, no events will result in the generation of system log messages.
Step 8	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 172.16.0.1`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 9	**path-option** *number* **explicit name** *name* **lockdown verbatim** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 explicit name explicit_path_a lockdown verbatim`	Specifies a path option for a headend router of a GMPLS tunnel. The path-option range is 1 to 1000.
Step 10	**path-option** *number* **explicit name** *name* **lockdown verbatim** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 20 explicit name explicit_path_b lockdown verbatim`	Specifies a path option. The path-option range is 1 to 1000.
Step 11	**path-option** *number* **no-ero lockdown** **Example:**	Specifies a path option with no ERO.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#` **`path-option 30 no-ero lockdown`**	
Step 12	**commit**	

Configuring and Verifying Multiple Path Options: Examples

This example shows how to configure multiple path options.

```
mpls traffic-eng
 gmpls optical-uni
   controller dwdm0/2/0/2
    tunnel-properties
     path-option 10 explicit name explicit_path_a lockdown verbatim
     path-option 20 explicit name explicit_path_b lockdown verbatim
     path-option 30 no-ero lockdown
    !
   !
```

The following sequence of examples show how to configure a GMPLS tunnel, add a new path option with a lower index than the path option in use, flap the tunnel and verify that the new path option (with a lower index) is used .

This example shows how to configure a GMPLS tunnel with one path option.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 10 explicit name explicit_path_a lockdown verbatim
   !
  !
 !
```

This example shows how to verify the tunnel path and status with a show command.

```
RP/0/RP0/CPU0:router#show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0  Destination: 172.16.0.1
  Signalled-Name: head_ot1001_172.16.0.1
GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN verbatim) type explicit explicit_path_a (Basis for Setup,
path weight 0)
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Fri Jul 17 08:41:21 ---- (3d06h ago)
...
```

This example shows how to add another path option with a lower index.

```
gmpls optical-uni
  controller dwdm0/3/0/0
```

```
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 1 no-ero lockdown
    path-option 10 explicit name explicit_path_a lockdown verbatim
   !
  !
 !
```

Flag the tunnel (or trigger reoptimization) and verify that the tunnel comes up on the path with a lower index.

```
RP/0/RP0/CPU0:router#show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0  Destination: 172.16.0.1
  Signalled-Name: head_ot1001_172.16.0.1
GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 1, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
    Last Signalled Error : Mon Jul 20 17:03:00 2015
      Info: [24] PathErr(2,2)-(Admin, reason unknown) at 50.0.0.2
    path option 10, (LOCKDOWN verbatim) type explicit explicit_all_loose_multi_hop
    Last Signalled Error : Mon Jul 20 17:03:00 ----
      Info: [25] PathErr(2,2)-(Admin, reason unknown) at 50.0.0.2
```

Enabling SRLG Discovery

Perform this task to enable SRLG discovery on the head node of a nLight tunnel.

Note

- SRLG discovery/recording is enabled only on the headend for each tunnel.
- SRLG discovery/recording allows a maximum of 62 SRLGs in RSVP, which is different from the maximum count of 64 in RSI.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **tunnel-properties**
6. **logging events lsp-status state**
7. **tunnel-id** *number*
8. **record srlg**
9. **destination ipv4 unicast** *address*
10. **path-option** *number* **no-ero lockdown**
11. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/2/0/0`	Enters GMPLS UNI controller submode.

Step 5	**tunnel-properties**
	Example:
	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`
		Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

Step 6	**logging events lsp-status state** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# logging events lsp-status state`	Configure events to generate system log messages when state changes occur on the GMPLS tunnel. If omitted, no events will result in the generation of system log messages.
Step 7	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 100`	Specifies a tunnel-ID for a headend router of a GMPLS tunnel. The tunnel-ID is a 16-bit number ranging from 0 to 65535.

	Command or Action	Purpose
Step 8	**record srlg** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# record srlg`	Enables SRLG recording.
Step 9	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 192.168.1.2`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 10	**path-option** *number* **no-ero lockdown** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 no-ero lockdown`	Specifies the path option for a headend router of a nLight (GMPLS) tunnel. The path-option range is 1 to 1000.
Step 11	**commit**	

Verifying SRLG Discovery Configuration: Examples

This example shows how to verify SRLG discovery configuration.

```
RP/0/0/CPU0:router#show mpls traffic-eng tunnels 100 detail

Name: GMPLS-UNI-dwdm0_2_0_0  Destination: 192.168.1.2
  Signalled-Name: rtrA_ot100_192.168.1.2
GMPLS UNI tunnel controlling link dwdm0/2/0/0, tunnel-id: 100
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Mon Jul 20 19:32:03 ---- (00:48:02 ago)
  Config Parameters:
    Priority:  7  7 Affinity: 0x0/0xffff
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Enabled
….
 Soft Preemption: None
    SRLGs: mandatory collection
    Path Info:
  ….
    Resv Info:
      Record Route:
        IPv4 10.10.10.2, flags 0x0
        SRLGs: 21, 22, 23, 24
      Fspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
Displayed 1 (of 3) heads, 0 (of 0) midpoints, 0 (of 2) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

This example shows how to verify SRLG discovery configuration at the headend and the tailend. The output shows the list of SRLGs.

```
RP/0/0/CPU0:router#show srlg

  System Information::
  Interface Count          : 4 (Maximum Interfaces Supported 512)
  Group Count              : 0 (Maximum Groups Supported 50)
  Inherit Location Count   : 0 (Maximum Inherit Locations Supported 10)
  Optical Interfaces Count : 4 (Maximum Optical Interfaces Supported 500)

  Interface    : GigabitEthernet0/2/0/0, Value Count : 10, Registrations : 2
  SRLG Values  : 11, 12, 13, 14, 15
                 21, 22, 23, 24, 25                                          -> Note:
These are announced srlgs.

  Interface    : GigabitEthernet0/2/0/2, Value Count : 0, Registrations : 1
  SRLG Values  :

  Interface    : GigabitEthernet0/2/0/4, Value Count : 0, Registrations : 2
  SRLG Values  :

  Interface    : GigabitEthernet0/2/0/5, Value Count : 0, Registrations : 2
  SRLG Values  :

  Optical Interface: dwdm0/2/0/0, Value Count : 5, References: 2
  SRLG Values      : 11, 12, 13, 14, 15                                       -> Note:
These are locally configured srlgs for controller (dwdm)

  Optical Interface: dwdm0/2/0/1, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/2, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/3, Value Count : 0, References: 1
  SRLG Values      :
```

Enabling SRLG Announce

Perform this task to enable SRLG announce. SRLG announce can be enabled on both headend and tailend.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **gmpls optical-uni**
4. **controller dwdm** *interface*
5. **announce srlgs**
6. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 4	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/1/0/1`	Enters GMPLS UNI controller submode.
Step 5	**announce srlgs** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# announce srlgs`	Announces discovered SRLGs to the system.
Step 6	**commit**	

Verifying SRLG Announce Configuration: Example

The following example shows how to verify SRLG announce configuration:

```
RP/0/0/CPU0:router#show srlg

  System Information::
  Interface Count          : 2 (Maximum Interfaces Supported 512)
  Group Count              : 0 (Maximum Groups Supported 50)
  Inherit Location Count   : 0 (Maximum Inherit Locations Supported 10)
  Optical Interfaces Count : 5 (Maximum Optical Interfaces Supported 500)

  Interface    : GigabitEthernet0/2/0/4, Value Count : 0, Registrations : 2
  SRLG Values  :
```

```
Interface    : GigabitEthernet0/2/0/5, Value Count : 0, Registrations : 2
SRLG Values  :

Interface: GigabitEthernet0/2/0/0, Value Count : 4, References: 1
SRLG Values      : 21, 22, 23, 24

Optical Interface: dwdm0/2/0/0, Value Count : 3, References: 2
SRLG Values      : 11, 12, 13

Optical Interface: dwdm0/2/0/1, Value Count : 0, References: 1
SRLG Values      :

Optical Interface: dwdm0/2/0/2, Value Count : 0, References: 1
SRLG Values      :

Optical Interface: dwdm0/2/0/3, Value Count : 0, References: 1
SRLG Values      :
```

Configuring SRLG Diversity

Perform this task to configure SRLG diversity with the best-effort or strict exclusion.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **attribute-set xro** *name*
4. **exclude [best-effort | strict] srlg value** *number*
5. **exit**
6. **gmpls optical-uni**
7. **controller dwdm** *interface*
8. **announce srlgs**
9. **tunnel-properties**
10. **logging events lsp-status state**
11. **tunnel-id** *number*
12. **record srlg**
13. **destination ipv4 unicast** *address*
14. **path-option** *number* **no-ero xro-attribute-set exclude_srlgs lockdown**
15. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters Global Configuration mode.

	Command or Action	Purpose
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**attribute-set xro** *name* **Example:** `RP/0/RSP0/CPU0:router(config-te)# attribute-set xro exclude_srlgs`	Enters the attribute set submode and specifies the attribute set name.
Step 4	**exclude [best-effort | strict] srlg value** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-attribute-set)# exclude best-effort srlg value 21`	Specifies path diversity based on SRLG.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-te-attribute-set)# exit`	Exits attribute-set submode.
Step 6	**gmpls optical-uni** **Example:** `RP/0/RSP0/CPU0:router(config-mpls-te)# gmpls optical-uni`	Enters GMPLS UNI configuration submode.
Step 7	**controller dwdm** *interface* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-uni)# controller dwdm 0/2/0/0`	Enters GMPLS UNI controller submode.
Step 8	**announce srlgs** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# announce srlgs`	Announces discovered SRLGs to the system.
Step 9	**tunnel-properties** **Example:**	Enters the submode to configure tunnel-specific information for a GMPLS UNI controller.

	Command or Action	Purpose

	`RP/0/RSP0/CPU0:router(config-te-gmpls-cntl)# tunnel-properties`
	
	`RP/0/RSP0/CPU0:router(config-te-gmpls-tun)#`	

Step 10	**logging events lsp-status state** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# logging events lsp-status state`	Configure events to generate system log messages when state changes occur on the GMPLS tunnel. If omitted, no events will result in the generation of system log messages.
Step 11	**tunnel-id** *number* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# tunnel-id 100`	Specifies a tunnel-ID for a headend router of a GMPLS tunnel. The tunnel-ID is a 16-bit number ranging from 0 to 65535.
Step 12	**record srlg** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# record srlg`	Enables SRLG recording.
Step 13	**destination ipv4 unicast** *address* **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# destination ipv4 unicast 192.168.1.2`	Specifies a tunnel destination for a headend router of a GMPLS tunnel. The destination argument is an IPv4 address.
Step 14	**path-option** *number* **no-ero xro-attribute-set exclude_srlgs lockdown** **Example:** `RP/0/RSP0/CPU0:router(config-te-gmpls-tun)# path-option 10 no-ero xro-attribute-set exclude_srlgs lockdown`	The XRO attribute set is attached to the GMPLS UNI tunnel through the path option. The path-option range is 1 to 1000.
Step 15	**commit**	

Verifying SRLG Diversity Configuration: Example

The following example shows how to verify SRLG diversity configuration:

```
RP/0/0/CPU0:router#show mpls traffic-eng tunnels 100 detail

Name: GMPLS-UNI-dwdm0_2_0_0  Destination: 192.168.1.2
  Signalled-Name: rtrA_ot100_192.168.1.2
GMPLS UNI tunnel controlling link dwdm0/2/0/0, tunnel-id: 100
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected
```

```
    path option 10, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
      XRO attribute-set: exclude_srlgs
        Best-effort, SRLG id 21
    Last Signalled Error : Mon Jul 20 20:55:33 ----
      Info: [5] PathErr(24,67)-(routing, route blocked by exclude route) at 10.10.10.2
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Mon Jul 20 19:32:03 2015 (01:25:19 ago)
  Config Parameters:
    Priority:  7  7 Affinity: 0x0/0xffff
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Enabled
  Binding Label: 0
  History:
    Tunnel has been up for: 00:00:23 (since Mon Jul 20 20:56:59 EDT 2015)
    Current LSP:
      Uptime: 00:00:23 (since Mon Jul 20 20:56:59 EDT ----)
  Current LSP Info:
    Instance: 6
    Uptime: 00:00:23 (since Mon Jul 20 20:56:59 EDT ----)
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 16
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 16
    Router-IDs: local      192.168.1.1
                downstream 192.168.1.2
    Soft Preemption: None
    SRLGs: mandatory collection
    Path Info:
      Outgoing:
        No ERO
      Route Exclusions:
        Best-effort, SRLG id 21
      Record Route: Disabled
      Tspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
      Session Attributes: Local Prot: Not Set, Node Prot: Not Set, BW Prot: Not Set
    Resv Info:
      Record Route:
        IPv4 10.10.10.2, flags 0x0
        SRLGs: 21, 22, 23, 24
      Fspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
Displayed 1 (of 3) heads, 0 (of 0) midpoints, 0 (of 2) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

Configuration Examples for GMPLS UNI

These configuration examples are provided for GMPLS UNI:

Configuring Head UNI-C for a GMPLS Tunnel: Example

This example shows the minimal head UNI-C configuration require to establish a GMPLS tunnel:

```
rsvp
  controller dwdm 0/1/0/1
    signalling refresh out-of-band interval 3600
    signalling refresh out-of-band missed 24
  !
!
mpls traffic-eng
  gmpls optical-uni
    controller dwdm 0/1/0/1
      tunnel-properties
        tunnel-id 100
        destination 100.20.20.20
        path-option 10 no-ero
      !
    !
  !
!
lmp
  gmpls optical-uni
    router-id 100.11.11.11
    neighbor nbr_A
      ipcc routed
      neighbor router-id ipv4 unicast 100.12.12.12
    !
    controller dwdm 0/1/0/1
      neighbor nbr_A
      link-id ipv4 unicast 192.168.100.1
      neighbor link-id ipv4 unicast 192.168.100.2
      neighbor interface-id unnumbered 13
    !
  !
!
```

Configuring Tail UNI-C for a GMPLS Tunnel: Example

This example shows the minimal tail UNI-C configuration require to establish a GMPLS tunnel:

Note The controller must be specified under the GMPLS UNI submode to inform TE that incoming GMPLS path messages are to be accepted and processed.

```
rsvp
  controller dwdm 0/1/0/1
    signalling refresh out-of-band interval 3600
    signalling refresh out-of-band missed 24
  !
!
mpls traffic-eng
  gmpls optical-uni
    controller dwdm 0/1/0/1
    !
  !
```

```
!
lmp
  gmpls optical-uni
    router-id 100.20.20.20
    neighbor nbr_B
      ipcc routed
      neighbor router-id ipv4 unicast 100.19.19.19
    !
    controller dwdm 0/1/0/1
      neighbor nbr_B
      link-id ipv4 unicast 192.168.103.2
      neighbor link-id ipv4 unicast 192.168.103.1
      neighbor interface-id unnumbered 22
    !
  !
!
```

Configuring LSP Diversity: Example

This example shows the configuration for two diverse LSPs:

```
  mpls traffic-eng
   attribute-set xro exclude-tun1
    exclude best-effort lsp source 88.0.0.8 destination 10.0.0.2 tunnel-id 1
extended-tunnel-id 88.0.0.8
    !
   attribute-set xro exclude-tun2
    exclude strict lsp source 88.0.0.8 destination 10.0.1.2 tunnel-id 2 extended-tunnel-id
 88.0.0.8 lsp-id 2
    !
   gmpls optical-uni
    controller dwdm 0/1/0/0
     tunnel-properties
      logging events lsp-status state
      tunnel-id 1
      destination ipv4 unicast 10.0.0.2
      path-option 10 no-ero xro-attribute-set exclude-tun2
     !
    !
    controller dwdm 0/1/0/1
     tunnel-properties
      logging events lsp-status state
      tunnel-id 2
      destination ipv4 unicast 10.0.1.2
      path-option 10 no-ero xro-attribute-set exclude-tun1
     !
    !
   !
  !
```

Additional References

For additional information related to implementing GMPLS UNI, refer to the following references:

Related Documents

Related Topic	Document Title
GMPLS UNI commands	*GMPLS UNI Commands* module in *MPLS Command Reference for Cisco ASR 9000 Series Routers*
MPLS Traffic Engineering commands	*MPLS Traffic Engineering commands* module in *MPLS Command Reference for Cisco ASR 9000 Series Routers*
RSVP commands	*RSVP commands* module in *MPLS Command Reference for Cisco ASR 9000 Series Routers*
Getting started material	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Information about user groups and task IDs	*Configuring AAA Services* module in *System Security Configuration Guide for Cisco ASR 9000 Series Routers*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 3471	*Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description*
RFC 3473	*Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions*
RFC 4208	*Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model*

RFCs	Title
RFC 4872	*RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery*
RFC 4874	*Exclude Routes - Extension to Resource ReserVation Protocol-Traffic Engineering (RSVP-TE)*
RFC 6205	*Generalized Labels for Lambda-Switch-Capable (LSC) Label Switching Routers*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport