Configuring IEEE 802.1Q Tunneling

This chapter describes how to configure IEEE 802.1Q tunneling on the Cisco 7600 series routers.

Note

- For complete syntax and usage information for the commands used in this chapter, refer to the Cisco 7600 Series Routers Command References at this URL:

  http://www.cisco.com/en/US/products/hw/routers/ps368/prod_command_reference_list.html
- The WS-X6548-GE-TX, WS-X6548V-GE-TX, WS-X6148-GE-TX, and WS-X6148V-GE-TX switching modules do not support IEEE 802.1Q tunneling.

This chapter contains these sections:

Understanding How 802.1Q Tunneling Works

802.1Q tunneling enables service providers to use a single VLAN to support customers who have multiple VLANs, while preserving customer VLAN IDs and keeping traffic in different customer VLANs segregated.

A port configured to support 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN that you dedicate to tunneling, which then becomes a tunnel VLAN. To keep customer traffic segregated, each customer requires a separate tunnel VLAN, but that one tunnel VLAN supports all of the customer's VLANs.

802.1Q tunneling is not restricted to point-to-point tunnel configurations. Any tunnel port in a tunnel VLAN is a tunnel entry and exit point. An 802.1Q tunnel can have as many tunnel ports as are needed to connect customer routers.

The customer routers are trunk connected, but with 802.1Q tunneling, the service provider routers only use one service provider VLAN to carry all the customer VLANs, instead of directly carrying all the customer VLANs.

With 802.1Q tunneling, tagged customer traffic comes from an 802.1Q trunk port on a customer device and enters the service-provider edge router through a tunnel port. The link between the 802.1Q trunk port on a customer device and the tunnel port is called an asymmetrical link because one end is configured as an 802.1Q trunk port and the other end is configured as a tunnel port. You assign the tunnel port to an access VLAN ID unique to each customer. See Figure 17-1 on page 17-2 and Figure 17-2 on page 17-3.

Figure 17-1 IEEE 802.1Q Tunnel Ports in a Service-Provider Network

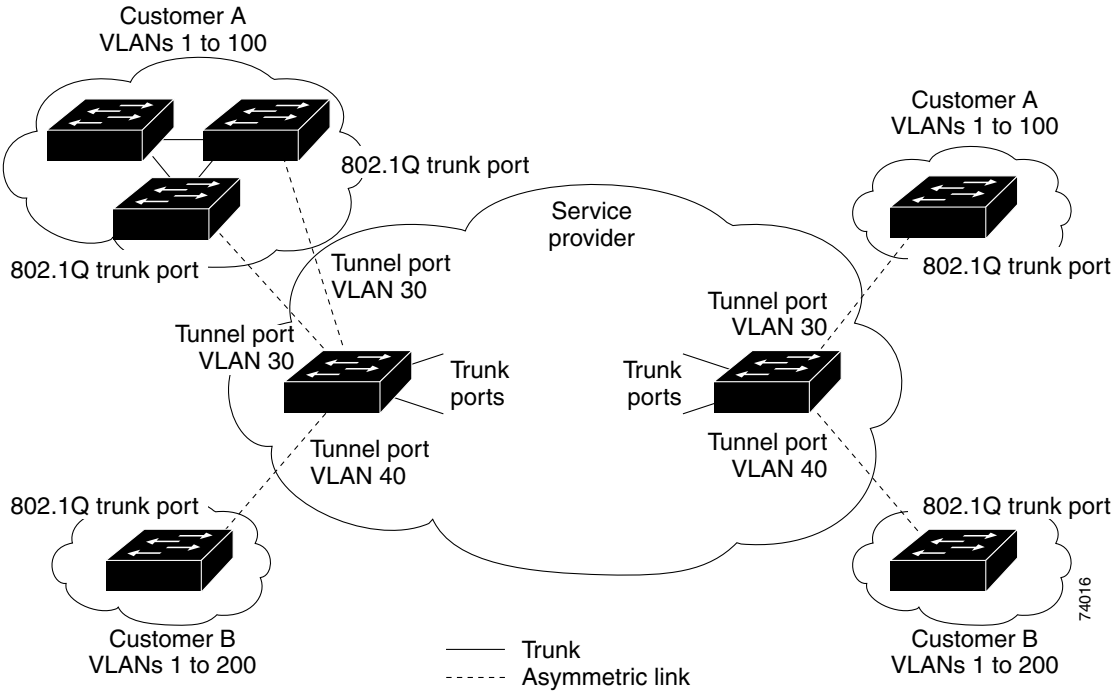
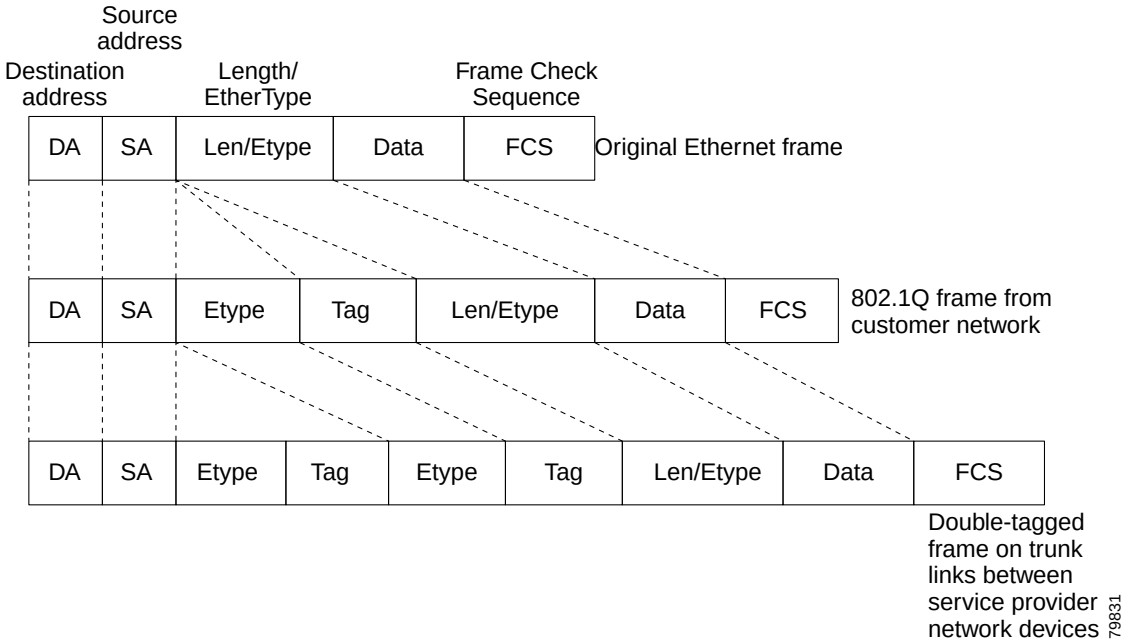


Figure 17-2 Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames



When a tunnel port receives tagged customer traffic from an 802.1Q trunk port, it does not strip the received 802.1Q tag from the frame header; instead, the tunnel port leaves the 802.1Q tag intact, adds a 2-byte Ethertype field (0x8100) followed by a 2-byte field containing the priority (CoS) and the VLAN. The received customer traffic is then put into the VLAN to which the tunnel port is assigned. This Ethertype 0x8100 traffic, with the received 802.1Q tag intact, is called tunnel traffic.

A VLAN carrying tunnel traffic is an 802.1Q tunnel. The tunnel ports in the VLAN are the tunnel's ingress and egress points.

The tunnel ports do not have to be on the same network device. The tunnel can cross other network links and other network devices before reaching the egress tunnel port. A tunnel can have as many tunnel ports as required to support the customer devices that need to communicate through the tunnel.

An egress tunnel port strips the 2-byte Ethertype field (0x8100) and the 2-byte length field and transmits the traffic with the 802.1Q tag still intact to an 802.1Q trunk port on a customer device. The 802.1Q trunk port on the customer device strips the 802.1Q tag and puts the traffic into the appropriate customer VLAN.

Note Tunnel traffic carries a second 802.1Q tag only when it is on a trunk link between service-provider network devices, with the outer tag containing the service-provider-assigned VLAN ID and the inner tag containing the customer-assigned VLAN IDs.

802.1Q Tunneling Configuration Guidelines and Restrictions

When configuring 802.1Q tunneling in your network, follow these guidelines and restrictions:

- Use asymmetrical links to put traffic into a tunnel or to remove traffic from a tunnel.
- Configure tunnel ports only to form an asymmetrical link.
- Dedicate one VLAN for each tunnel.
- Assign only tunnel ports to VLANs used for tunneling.
- Trunks require no special configuration to carry tunnel VLANs.
- Tunnel ports are not trunks. Any commands to configure trunking are inactive while the port is configured as a tunnel port.
- Tunnel ports learn customer MAC addresses.
- We recommend that you use ISL trunks to carry tunnel traffic between devices that do not have tunnel ports. Because of the 802.1Q native VLAN feature, using 802.1Q trunks requires that you be very careful when you configure tunneling: a mistake might direct tunnel traffic to a non-tunnel port.
- Ensure that the native VLAN of the 802.1Q trunk port in an asymmetrical link carries no traffic. Because traffic in the native VLAN is untagged, it cannot be tunneled correctly. Alternatively, you can enter the global **vlan dot1q tag native** command to tag native VLAN egress traffic and drop untagged native VLAN ingress traffic.
- Configure jumbo frame support on tunnel ports:
  - See the "Configuring Jumbo Frame Support" section on page 8-8.
  - Take note of the modules listed in the "Configuring Jumbo Frame Support" section that do not support jumbo frames.
- Jumbo frames can be tunneled as long as the jumbo frame length combined with the 802.1Q tag does not exceed the maximum frame size.
- Because tunnel traffic has the added ethertype and length field and retains the 802.1Q tag within the router, the following restrictions exist:
  - The Layer 3 packet within the Layer 2 frame cannot be identified in tunnel traffic.
  - Layer 3 and higher parameters cannot be identified in tunnel traffic (for example, Layer 3 destination and source addresses).
  - Because the Layer 3 addresses cannot be identified within the packet, tunnel traffic cannot be routed.
  - The router can provide only MAC-layer filtering for tunnel traffic (VLAN IDs and source and destination MAC addresses).
  - The router can provide only MAC-layer access control and QoS for tunnel traffic.
  - QoS cannot detect the received CoS value in the 802.1Q 2-byte Tag Control Information field.
- On an asymmetrical link, the Cisco Discovery Protocol (CDP) reports a native VLAN mismatch if the VLAN of the tunnel port does not match the native VLAN of the 802.1Q trunk. The 802.1Q tunnel feature does not require that the VLANs match. Ignore the messages if your configuration requires nonmatching VLANs.
- Asymmetrical links do not support the Dynamic Trunking Protocol (DTP) because only one port on the link is a trunk. Configure the 802.1Q trunk port on an asymmetrical link to trunk unconditionally.

- The 802.1Q tunneling feature cannot be configured on ports configured to support private VLANs.
- The following Layer 2 protocols work between devices connected by an asymmetrical link:
  - CDP
  - UniDirectional Link Detection (UDLD)
  - Port Aggregation Protocol (PAgP)
  - Link Aggregation Control Protocol (LACP)
- PortFast BPDU filtering is enabled automatically on tunnel ports.
- CDP is automatically disabled on tunnel ports.
- VLAN Trunk Protocol (VTP) does not work between the following devices:
  - Devices connected by an asymmetrical link
  - Devices communicating through a tunnel

Note VTP works between tunneled devices if Layer 2 protocol tunneling is enabled. See Chapter 18, "Configuring Layer 2 Protocol Tunneling," for configuration details.

- To configure an EtherChannel as an asymmetrical link, all ports in the EtherChannel must have the same tunneling configuration. Because the Layer 3 packet within the Layer 2 frame cannot be identified, you must configure the EtherChannel to use MAC-address-based frame distribution.

The following configuration guidelines are *required* for your Layer 2 protocol tunneling configuration:

- On all the service provider edge routers, PortFast BPDU filtering must be enabled on the 802.1Q tunnel ports as follows:

```
Router(config-if)# spanning-tree bpdufilter enable
Router(config-if)# spanning-tree portfast
```

Note PortFast BPDU filtering is enabled automatically on tunnel ports.

- At least one VLAN must be available for Native VLAN tagging (**vlan dot1q tag native** option). If you use all the available VLANs and then try to enable the **vlan dot1q tag native** option, the option will not be enabled.
- On all the service provider core routers, tag native VLAN egress traffic and drop untagged native VLAN ingress traffic by entering the following command:

```
Router(config)# vlan dot1q tag native
```

- On all the customer routers, *either* enable or disable the global **vlan dot1q tag native** option.

Note If this option is enabled on one router and disabled on another router, all traffic is dropped; all customer routers must have this option configured the same on each router.

The following configuration guidelines are *optional* for your Layer 2 protocol tunneling configuration:

- Because all the BPDUs are being dropped, spanning tree PortFast can be enabled on Layer 2 protocol tunnel ports as follows:

```
Router(config-if)# spanning-tree portfast trunk
```

- If the service provider does not want the customer to see its routers, CDP should be disabled on the 802.1Q tunnel port as follows:

```
Router(config-if)# no cdp enable
```

Configuring 802.1Q Tunneling

These sections describe 802.1Q tunneling configuration:

- Configuring 802.1Q Tunnel Ports, page 17-6
- Configuring the Router to Tag Native VLAN Traffic, page 17-7

Caution Ensure that only the appropriate tunnel ports are in any VLAN used for tunneling and that one VLAN is used for each tunnel. Incorrect assignment of tunnel ports to VLANs can forward traffic inappropriately.

Configuring 802.1Q Tunnel Ports

To configure 802.1Q tunneling on a port, perform this task:

	Command	Purpose
Step 1	`Router(config)# interface` *type*[1] *slot/port*	Selects the LAN port to configure.
Step 2	`Router(config-if)# switchport`	Configures the LAN port for Layer 2 switching: • You must enter the **switchport** command once without any keywords to configure the LAN port as a Layer 2 interface before you can enter additional **switchport** commands with keywords. • Required only if you have not entered the **switchport** command already for the interface.
Step 3	`Router(config-if)# switchport mode dot1q-tunnel`	Configures the Layer 2 port as a tunnel port.
	`Router(config-if)# no switchport mode dot1q-tunnel`	Clears the tunnel port configuration.
Step 4	`Router(config-if)# end`	Exits configuration mode.
Step 5	`Router# show dot1q-tunnel [{interface` *type interface-number*`}]`	Verifies the configuration.

1. *type* = **ethernet**, **fastethernet**, **gigabitethernet**, or **tengigabitethernet**

This example shows how to configure tunneling on port 4/1 and verify the configuration:

```
Router# configure terminal
Router(config)# interface fastethernet 4/1
Router(config-if)# switchport mode dot1q-tunnel
Router(config-if)# end
Router# show dot1q-tunnel interface
```

Configuring the Router to Tag Native VLAN Traffic

The **vlan dot1q tag native** command is a global command that configures the router to tag native VLAN traffic, and admit only 802.1Q tagged frames on 802.1Q trunks, dropping any untagged traffic, including untagged traffic in the native VLAN.

Note Staring from Release 15.1(2)S, if the native VLAN tagging is enabled globally using the **vlan dot1q tag native command**, or locally on the port using the **switchport trunk native vlan command**, then the UDLD packets are sent with VLAN 1 from L2 ports.

To configure the router to tag traffic in the native VLAN, perform this task:

	Command	Purpose
Step 1	`Router(config)# vlan dot1q tag native`	Configures the router to tag native VLAN traffic.
	`Router(config)# no vlan dot1q tag native`	Clears the configuration.
Step 2	`Router(config)# end`	Exits configuration mode.
Step 3	`Router# show vlan dot1q tag native`	Verifies the configuration.

This example shows how to configure the router to tag native VLAN traffic and verify the configuration:

```
Router# configure terminal
Router(config)# vlan dot1q tag native
Router(config)# end
Router# show vlan dot1q tag native
```

IEEE 802.1ab LLDP

The following sections describe the IEEE 802.1ab Link Layer Discovery Protocol (LLDP) function.

Understanding LLDP

The Cisco Discovery Protocol (CDP) is a layer 2 device discovery protocol (the data link layer) on all Cisco-manufactured devices (routers, bridges, access servers, and switches). CDP allows network management applications to automatically discover other Cisco devices connected to the network.

For more information on CDP commands, see the Cisco IOS Configuration Fundamentals Command Reference Guide at the following location:

http://www.cisco.com/en/US/docs/ios/12_1/configfun/command/reference/frd3001b.html

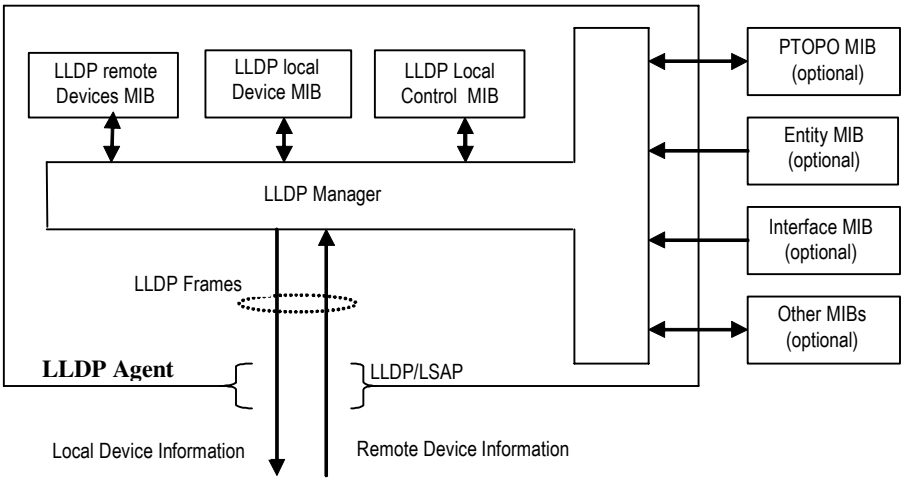
CDP is mostly used for topology discovery. A non-Cisco device cannot interact with a Cisco device using CDP. To support non-Cisco devices and to allow interoperability between other devices, the IEEE 802.1AB LLDP is used.

These are some LLDP features:

- The protocol allows two systems (running different network layer protocols) to learn about each other.
- LLDP is an optional element of a protocol stack in the 802 LAN station.
- LLDP uses the logical link control (LLC) services to transmit and receive information to and from other LLDP agents. LLC provides a Link Service Access Point(LSAP) for access to LLDP. Each LLDP frame is transmitted as a single MAC Service request. Each incoming LLDP frame is received at the MAC Service Access Point(MSAP) by the LLC entity as a MAC service indication.

Figure 17-3 shows a high-level view of LLDP operating in a network.

Figure 17-3 LLDP Block Diagram



The LLDP protocol operates through the LLDP Agent. The tasks of the LLDP agent are to:

- Collect information from the LLDP local system MIB and transmit it periodically.
- Receive LLDP frames from neighbors and populate LLDP remote devices MIB and other optional MIBs.

LLDP supports a set of attributes used to find the neighbor devices. These attributes are type, length, and value descriptions of devices, and are referred to as Type Length Value (TLV). LLDP supported devices use TLVs to receive and send information to their neighbors. Details such as configuration information, device capabilities, and device identity are also advertised using this protocol.

The mandatory LLDP TLVs are:

- Port description
- System name
- System description
- System capabilities
- Management address

Restrictions

The following restrictions apply to LLDP:

- The memory available on a given end network device dictates the number of neighbor entries recorded. However, under most operating conditions, end devices such as printers, IP phones, workstations and so on, are typically operated in the receive mode only.
- If objects from the Entity MIB are used for LLDP broadcast, such as to create a sender ID, these MIBs should be available before LLDP can function correctly.

Default LLDP Configuration

Table 17-1 shows the default LLDP configuration.

Table 17-1 Default LLDP Configuration

Feature	Default Setting
LLDP global state	Disabled
LLDP holdtime (before discarding)	120 seconds
LLDP timer (packet update frequency)	30 seconds
LLDP reinitialization delay	2 seconds
LLDP tlv-select	Enabled to send and receive all TLVs.
LLDP interface state	Enabled
LLDP receive	Enabled
LLDP transmit	Enabled

Configuring LLDP on the Cisco 7600 series router

The following sections describe how to configure LLDP on the c7600 platform:

- Configuring LLDP, page 17-9
- Monitoring and Maintaining LLDP, page 17-12
- Verifying the Configuration, page 17-13

Configuring LLDP

Complete the following steps to configure LLDP.

SUMMARY STEPS

Step 1 **enable**

Step 2 **configure terminal**

Step 3 **lldp** {**run** | **holdtime** *seconds*| **reinit** | **timer** *rate*| **tlv-select**}

Step 4 **end**

DETAILED STEPS

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

	Command or Action	Purpose

Step 3	**lldp run**
	
	Example:
	`Router(config)# lldp run`
	
	or
	
	lldp holdtime *seconds*
	
	Example:
	`Router(config)# lldp holdtime 100`
	
	or
	
	lldp reinit
	
	Example:
	`Router(config)# lldp reinit 2`
	
	or
	
	lldp timer *rate*
	
	Example:
	`Router(config)# lldp timer 75`
	
	or
	
	lldp tlv-select
	
	Example:
	`Router(config-if)# lldp tlv-select system-description`
		Enables LLDP globally on all the interfaces on the router.
		
		Specifies the hold time. The value ranges from 0 to 65535 seconds. The default value is 120 seconds.
		
		Specifies the delay time in seconds for LLDP to initialize on any interface.
		
		The value ranges from 2 to 5 seconds. The default value is 2 seconds.
		
		Specifies the rate at which LLDP packets are sent. The value ranges from 5 to 65534 seconds. The default value is 30 seconds.
		
		Enables a specific LLDP TLV on a supported interface,
		• **management-address:** Specifies the management address TLV
		• **port-description**: Specifies the port description TLV
		• **system-capabilities**: Specifies the system capabilities TLV
		• **system-description**: Specifies the system description TLV
		• **system-name**: Specifies the system name TLV

Step 4	**end** **Example:** `Router(config-if)# end`	Returns the CLI to privileged EXEC mode.

Configuration Examples

This is an example to enable LLDP globally.

```
Router> enable
Router# configure terminal
Router(config)# lldp run
Router(config)# end
```

This is an example to define a hold time for an LLDP-enabled device.

```
Router> enable
Router# configure terminal
```

```
Router(config)# lldp holdtime 100
Router(config)# end
```

This is an example to specify the delay time in seconds for LLDP to initialize:

```
Router> enable
Router# configure terminal
Router(config)# lldp reinit 2
Router(config)# end
```

This is an example to specify an interval at which the Cisco IOS software sends LLDP updates to the neighboring devices.

```
Router> enable
Router# configure terminal
Router(config)# lldp timer 75
Router(config)# end
```

This is an example to enable an LLDP TLV on a supported interface:

```
Router> enable
Router# configure terminal
Router(config)# interface ethernet 0/1
Router(config-if)# lldp tlv-select system-description
Router(config-if)# end
```

Monitoring and Maintaining LLDP

Complete the following steps to monitor and maintain LLDP.

SUMMARY STEPS

Step 1 **enable**

Step 2 **show lldp** [**entry** {***** | *word*} | **errors** | **interface** [**ethernet** *number*] | **neighbors** [**ethernet** *number* | **detail**] | **traffic**]

Step 3 **clear lldp** {**counters** | **table**}

Step 4 **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show lldp** [**entry** {***** | *word*} | **errors** | **interface** [**ethernet** *number*] | **neighbors** [**ethernet** *number* | **detail**] | **traffic**] **Example:** `Router# show lldp entry *`	Displays LLDP information. • **entry**: Specifies the information for specific neighbor entry • **errors**: Specifies the LLDP computational errors and overflows • **interface**: Specifies the LLDP interface status and configuration • **neighbors**: Specifies the LLDP neighbor entries • **traffic**: Specifies the LLDP statistics • |: Specifies the output modifiers **Note** When you use the **show lldp neighbors** command, and if the device ID has more than 20 characters, the ID is truncated to 20 characters in command output because of display constraints.
Step 3	**clear lldp {counters | table}** **Example:** `Router# clear lldp counters`	Resets LLDP traffic counters and tables to zero.
Step 4	**end** **Example:** `Router# end`	Returns the CLI to user EXEC mode.

Configuration Examples

This is an example to monitor and maintain LLDP:

```
Router> enable
Router# show lldp entry *
Router# clear lldp counters
Router# end
```

Verifying the Configuration

This section provides the commands to verify the configuration of LLDP on the Cisco 7600 series router:

```
Router# show lldp ?
      entry      Information for specific neighbor entry
      errors     LLDP computational errors and overflows
      interface  LLDP interface status and configuration
      neighbors  LLDP neighbor entries
      traffic    LLDP statistics
      |          Output modifiers
```

```
      <cr>

Router# show lldp entry *

    Capability codes:
        (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
        (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
    ------------------------------------------------
    Chassis id: 001b.0de4.9040
    Port id: Te5/4
    Port Description: TenGigabitEthernet5/4
    System Name: RSP

    System Description:
    Cisco IOS Software, rsp72043_rp Software (rsp72043_rp-ADVENTERPRISEK9_DBG-M),
    Experimental Version 15.1(20101020:182513) [mcp_dev-kalnaray-lldp-mcp-lds 202]
    Copyright (c) 1986-2010 by Cisco Systems, Inc.
    Compiled Sun 24-Oct-10 23:59 by kalnaray

    Time remaining: 113 seconds
    System Capabilities: B,R
    Enabled Capabilities: R
    Management Addresses:
        IP: 182.0.0.1
    Auto Negotiation - not supported
    Physical media capabilities - not advertised
    Media Attachment Unit type - not advertised
    Vlan ID: - not advertised
    Total entries displayed: 1

Router# show lldp errors

    LLDP errors/overflows:
        Total memory allocation failures: 0
        Total encapsulation failures: 0
        Total input queue overflows: 0
        Total table overflows: 0

Router# show lldp interface

    TenGigabitEthernet1/1:
        Tx: enabled
        Rx: enabled
        Tx state: IDLE
        Rx state: WAIT FOR FRAME

    TenGigabitEthernet1/2:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER

    TenGigabitEthernet1/3:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER

    TenGigabitEthernet1/4:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER
```

```
    GigabitEthernet5/1:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER

    GigabitEthernet5/2:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER

    Control Plane:
        Tx: enabled
        Rx: enabled
        Tx state: INIT
        Rx state: WAIT PORT OPER

Router# show lldp neighbors
    Capability codes:
        (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
        (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other

    Device ID           Local Intf     Hold-time  Capability      Port ID
    RSP                 Te1/1          120        R               Te5/4

    Total entries displayed: 1

Router# show lldp traffic

    LLDP traffic statistics:
        Total frames out: 8
        Total entries aged: 0
        Total frames in: 4
        Total frames received in error: 0
        Total frames discarded: 0
        Total TLVs discarded: 0
        Total TLVs unrecognized: 0

Router# clear lldp ?
      counters  Clear LLDP counters
      table     Clear lldp table

Router# show lldp

    Global LLDP Information:
        Status: ACTIVE
        LLDP advertisements are sent every 30 seconds
        LLDP hold time advertised is 120 seconds
        LLDP interface reinitialisation delay is 2 seconds
```

Troubleshooting Tips

Scenarios/Problems	Solution

How do I collect debugging information on LLDP errors?	Use the **debug lldp errors** command. This is a sample output:
	`Router# debug lldp errors`
	`LLDP errors debugging is on`

How do I collect details on all events occurring on LLDP?	Use the **debug lldp events** command. This is a sample output:
	`Router# debug lldp events`
	`LLDP events debugging is on`

How do I collect packet-related debugging information on an LLDP?	Use the **debug lldp packet** command. This is a sample output:
	`Router# debug lldp packet`
	`LLDP packet info debugging is on`

How do I collect debugging information on state change, when there is a switchover?	Use the **debug lldp state** command. This is a sample output:
	`Router# debug lldp state`
	`LLDP state debugging is on`

How do I verify the debug commands for LLDP configuration?	Use the **show debugging** command. This is a sample output:
	`Router# show debugging`
	`LLDP:`
	`  LLDP packet info debugging is on`
	`  LLDP events debugging is on`
	`  LLDP errors debugging is on`
	`  LLDP states debugging is on`