L2TPv3: Layer 2 Tunnel Protocol Version 3

The Layer 2 Tunnel Protocol Version 3 feature expands on Cisco support of the Layer 2 Tunnel Protocol Version 3 (L2TPv3). L2TPv3 is an Internet Engineering Task Force (IETF) l2tpext working group draft that provides several enhancements to L2TP for the capability to tunnel any Layer 2 payload over L2TP. Specifically, L2TPv3 defines the L2TP protocol for tunneling Layer 2 payloads over an IP core network using Layer 2 virtual private networks (VPNs). Benefits of this feature include the following:

- L2TPv3 simplifies deployment of VPNs
- L2TPv3 does not require Multiprotocol Label Switching
- L2TPv3 supports Layer 2 tunneling over IP for any payload

Feature History for Layer 2 Tunnel Protocol Version 3

Release	Modification
12.0(21)S	Initial data plane support for L2TPv3 was introduced on the Cisco 7200 series, Cisco 7500 series, Cisco 10720, and Cisco 12000 series platforms.
12.0(23)S	L2TPv3 control plane support was introduced on the Cisco 7200 series, Cisco 7500 series, Cisco 10720, and Cisco 12000 series platforms.
12.3(2)T	This feature was integrated into Cisco IOS Release 12.3(2)T and implemented on the Cisco 2600XM series Multiservice platforms, the Cisco 2691 Multiservice routers, the Cisco 3662 Multiservice Access platforms, the Cisco 3725 Modular Access routers, and the Cisco 3745 Modular Access routers.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Contents

Prerequisites for Layer 2 Tunnel Protocol Version 3

- Before you configure an Xconnect attachment circuit for a customer edge (CE) device (see the section "Configuring the Xconnect Attachment Circuit"), the Cisco Express Forwarding (CEF) feature must be enabled. To enable CEF on an interface, use the **ip cef** or **ip cef distributed** command.
- You must configure a loopback interface on the router for originating and terminating the L2TPv3 traffic. The loopback interface must have an IP address that is reachable from the remote provider edge (PE) device at the other end of an L2TPv3 control channel.
- To enableSimple Network Management Protocol (SNMP) notifications of L2TP session up and down events, enter the **snmp-server enable traps l2tun session** command before configuring L2TPv3.

Restrictions for Layer 2 Tunnel Protocol Version 3

The following subsections contain information on restrictions:

- Supported Port Adapters for the Cisco 7200 and 7500 Series Routers
- General L2TPv3 Restrictions
- Cisco 7500-Specific Restrictions
- Frame Relay-Specific Restrictions
- VLAN-Specific Restrictions

Supported Port Adapters for the Cisco 7200 and 7500 Series Routers

L2TPv3 is supported on the following port adapters in the Cisco 7200 and 7500 series routers:

- Single-port Fast Ethernet 100BASE-TX
- Single-port Fast Ethernet 100BASE-FX
- Dual-port Fast Ethernet 100BASE-TX
- Dual-port Fast Ethernet 100BASE-FX
- Gigabit Ethernet port adapter

- 12-port Ethernet/2-port FE adapter
- 4-port synchronous serial port adapter
- Enhanced 4-port synchronous serial port adapter
- 8-port synchronous serial port adapter
- Single-port HSSI adapter
- Dual-port HSSI adapter
- 8-port multichannel E1 G.703/G.704 120-ohm interfaces
- 2-port multichannel E1 G.703/G.704 120-ohm interfaces
- 8-port multichannel T1 with integrated DSUs
- 8-port multichannel T1 with integrated CSUs and DSUs
- 4-port multichannel T1 with integrated CSUs and DSUs
- 2-port multichannel T1 with integrated CSUs and DSUs
- 8-port multichannel T1/E1
- 1-port multichannel T3 interface
- 1-port multichannel E3 interface
- 2-port enhanced multichannel T3 port adapter
- Single-port T3 port adapter
- Single-port E3 port adapter
- 2-port T3 port adapter
- 2-port T3 port adapter
- Single-port PoS, single-mode, long reach
- Single-port PoS, single-mode, intermediate reach
- Single-port PoS, multimode

L2TPv3 is supported on the following port adapters for the Cisco 7200 series routers only:

- 8-port Ethernet adapter
- 4-port Ethernet adapter

General L2TPv3 Restrictions

- Cisco Express Forwarding (CEF) must be enabled for the L2TPv3 feature to function. The Xconnect configuration submode is blocked until CEF is enabled. On distributed platforms, such as the Cisco 7500 series, if CEF is disabled while a session is established, the session is torn down and remains down until CEF is reenabled. To enable CEF, use the **ip cef** or **ip cef distributed** command.
- Specifying a source IP address to configure a loopback interface is highly recommended. If you do not configure a loopback interface, the router will choose the best available local address, which could be any IP address configured on a core-facing interface. This configuration could prevent a control channel from being established
- The number of sessions on a PPP, high-level data link control (HDLC), Ethernet, or 802.1q VLAN port is limited by the number of interface descriptor blocks (IDBs) that the router can support. For PPP, HDLC, Ethernet, and 802.1q VLAN circuit types, an IDB is required for each circuit.

When L2TPv3 is used to tunnel Frame Relay data-link connection identifiers (DLCIs), an IDB is not required for each circuit. As a result, the memory requirements are much lower. The scalability targets for the Engineering Field Test (EFT) program are 4000 L2TP sessions.

- Frame Relay support includes only 10-bit DLCI addressing. The L2TPv3 feature does not support Frame Relay extended addressing.
- The interface keepalive feature is automatically disabled on the interface to which Xconnect is applied, except for Frame Relay encapsulation, which is required for the Local Management Interface (LMI).
- Static L2TPv3 sessions do not support Frame Relay LMI interworking.
- Static L2TPv3 sessions do not interoperate with Universal Transport Interface (UTI) using keepalives.
- The **ip pmtu** command used to configure the pseudowire class (see the section "Configuring the L2TPv3 Pseudowire") is not supported for static L2TPv3 sessions. As a result, Intermediate System-to-Intermediate System (IS-IS) fragmentation through a static L2TPv3 session is not supported.

Cisco 7500-Specific Restrictions

- Although L2TPv3 sequencing is supported on Cisco 7500 series routers, all L2TP packets that require sequence number processing will be sent to the Route/Switch Processor (RSP) module.
- Distributed CEF (dCEF) is not supported with sequencing.

Frame Relay-Specific Restrictions

- Frame Relay per-DLCI forwarding and port-to-port trunking are mutually exclusive. L2TPv3 does not support the use of both on the same interface at the same time.
- The **xconnect** command is not supported on Frame Relay interfaces directly. For Frame Relay, the Xconnect is applied under the **connect** command specifying the DLCI to be used.
- Changing the encapsulation type on any interface removes any existing **xconnect** command applied to that interface.
- The discard eligible (DE) bit value does not influence the type of service (ToS) bits.
- To use DCE or a Network-to-Network Interface (NNI) on a Frame Relay port, you must configure the **frame-relay switching** command.
- Frame Relay policing is nondistributed on the Cisco 7500 series. By configuring Frame Relay policing, you cause traffic on the affected PVCs to be sent to the RSP for processing. Frame Relay policing is not supported on the Cisco 12000 series Internet router.
- Frame Relay support is for 10-bit DLCI addresses. Frame Relay Extended Addressing is not supported.
- Multipoint DLCI is not supported.
- The keepalive will automatically be disabled on interfaces that have an Xconnect applied to them, except for Frame Relay encapsulation, which is a requirement for LMI.
- Static L2TPv3 sessions will not support Frame Relay LMI interworking.

VLAN-Specific Restrictions

- A provider edge (PE) router is responsible only for static VLAN membership entries that are manually configured on the router. Dynamic VLAN membership entries, entry aging, and membership discovery are not supported.

- Implicit tagging for VLAN membership operating on the other layers (such as at Layer 2, membership by MAC address or protocol type, at Layer 3, or membership by IP subnet) is not supported.
- Point-to-multipoint and multipoint-to-point configurations are not supported. There is a 1:1 relationship between an attachment circuit and an L2TPv3 session.

Information About Layer 2 Tunnel Protocol Version 3

To configure the Layer 2 Tunnel Protocol Version 3 feature, you must understand the following concepts:

Migration from UTI to L2TPv3

UTI is a Cisco proprietary protocol that offers a simple high-speed transparent Layer 2-to-Layer 2 service over an IP backbone. The UTI protocol lacks the signaling capability and standards support necessary for large-scale commercial service. To begin to answer the need for a standard way to provide large-scale VPN connectivity over an IP core network, limited migration from UTI to L2TPv3 was introduced in Cisco IOS Release 12.0(21)S. The L2TPv3 feature in Cisco IOS Release 12.0(23)S introduced a more robust version of L2TPv3 to replace UTI. This more robust version of L2TPv3 is now available in Cisco IOS Release 12.3(2)T.

As described in the section "L2TPv3 Header Description," the UTI data header is identical to the L2TPv3 header but with no sequence numbers and an 8-byte cookie. By manually configuring an L2TPv3 session using an 8-byte cookie (see the section "Manually Configuring L2TPv3 Session Parameters") and by setting the IP protocol number of outgoing data packets to 120 (as described in the section "Configuring the L2TPv3 Pseudowire"), you can ensure that a PE running L2TPv3 may interoperate with a peer PE running UTI. However, because UTI does not define a signaling plane, dynamically established L2TPv3 sessions cannot interoperate with UTI.

When a customer upgrades from a pre-L2TPv3 Cisco IOS release to a post-L2TPv3 release, an internal UTI-to-Xconnect command-line interface (CLI) migration utility will automatically convert the UTI commands to Xconnect and pseudowire class configuration commands without the need for any user intervention. After the CLI migration, the UTI commands that were replaced will not be available. The old-style UTI CLI will be hidden from the user.

Note The UTI keepalive feature will *not* be migrated. The UTI keepalive feature will no longer be supported in post-L2TPv3 releases. You should convert to using dynamic L2TPv3 sessions in order to preserve the functionality provided by the UTI keepalive.

L2TPv3 Operation

L2TPv3 provides similar and enhanced services to replace the current UTI implementation, including the following features:

- Xconnect for Layer 2 tunneling via a pseudowire over an IP network
- Layer 2 VPNs for PE-to-PE router service via Xconnect that support Ethernet, 802.1q (VLAN), Frame Relay, HDLC and PPP Layer 2 circuits, including both static (UTI-like) and dynamic (using the new L2TPv3 signaling) forwarded sessions

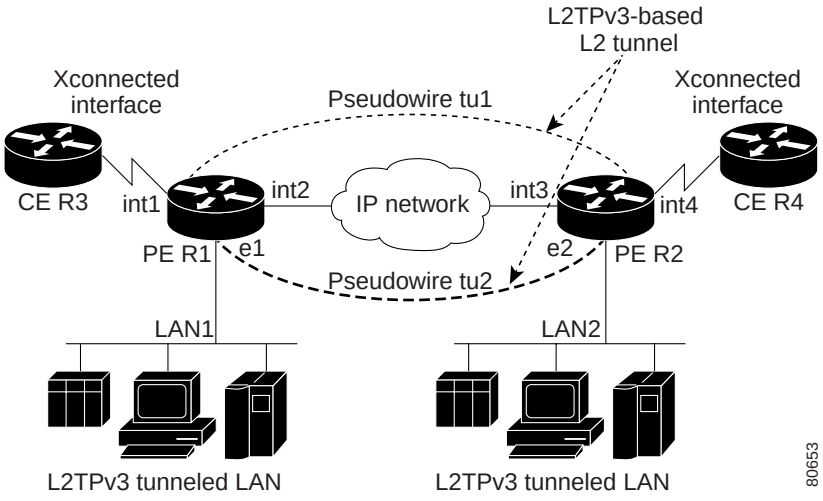
The Cisco IOS Release 12.3(2)T feature supports only the following features:

- Layer 2 tunneling (as used in an L2TP access concentrator, or LAC) to an attachment circuit, not Layer 3 tunneling
- L2TPv3 data encapsulation directly over IP (IP protocol number 115), not using User Datagram Protocol (UDP)
- Point-to-point sessions, not point-to-multipoint or multipoint-to-point sessions
- Sessions between the same Layer 2 protocols; for example, Ethernet-to-Ethernet, VLAN-to-VLAN, but not VLAN-to-Ethernet or Frame Relay

The attachment circuit is the physical interface or subinterface attached to the pseudowire.

Figure 1 shows an example of how the L2TPv3 feature is used for setting up VPNs using Layer 2 tunneling over an IP network. All traffic between two customer network sites is encapsulated in IP packets carrying L2TP data messages and sent across an IP network. The backbone routers of the IP network treat the traffic as any other IP traffic and need not know anything about the customer networks.

Figure 1 L2TPv3 Operation



In Figure 1, the PE routers R1 and R2 provide L2TPv3 services. The R1 and R2 routers communicate with each other using a pseudowire over the IP backbone network through a path comprising the interfaces int1 and int2, the IP network, and interfaces int3 and int4.

In this example, the CE routers R3 and R4 communicate through a pair of Xconnect Ethernet or 802.1q VLAN interfaces using an L2TPv3 session. The L2TPv3 session tu1 is a pseudowire configured between interface int1 on R1 and interface int4 on R2. Any packet arriving on interface int1 on R1 is encapsulated and sent via the pseudowire control channel (tu1*)* to R2. R2 decapsulates the packet and sends it on interface int4 to R4. When R4 needs to send a packet to R3, the packet follows the same path in reverse.

Please note the following features regarding L2TPv3 operation:

- All packets received on interface int1 will be forwarded to R4. R3 and R4 cannot detect the intervening network.
- For Ethernet interfaces, any packet received from LAN1 by R1 on Ethernet interface e1 will be encapsulated directly in IP and sent via the pseudowire session tu2 to R2 interface e2, where it will be sent on LAN2.
- A VLAN on an Ethernet interface can be mapped to an L2TPv3 session.

Benefits of Using L2TPv3

L2TPv3 Simplifies Deployment of VPNs

L2TPv3 is an industry-standard Layer 2 tunneling protocol that ensures interoperability among vendors, increasing customer flexibility and service availability.

L2TPv3 Does Not Require MPLS

With L2TPv3 service providers need not deploy MPLS in the core IP backbone to set up VPNs using L2TPv3 over the IP backbone, resulting in operational savings and increased revenue.

L2TPv3 Supports Layer 2 Tunneling over IP for Any Payload

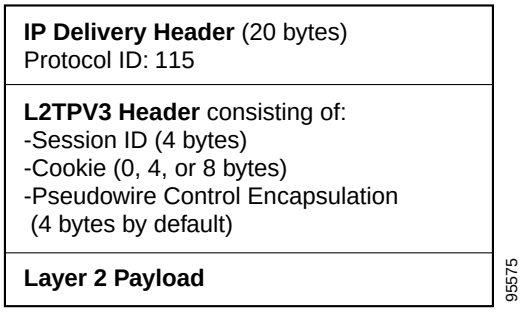
L2TPv3 provides enhancements to L2TP to support Layer 2 tunneling of any payload over an IP core network. L2TPv3 defines the base L2TP protocol as being separate from the Layer 2 payload that is tunneled.

L2TPv3 Header Description

The migration from UTI to L2TPv3 also requires the standardization of the UTI header. As a result, the L2TPv3 header has the new format shown in Figure 2.

Each L2TPv3 packet contains an L2TPv3 header that includes a unique session ID representing one session and a variable cookie length. The L2TPv3 session ID and the Tunnel Cookie field length are assigned via the CLI. See the section "How to Configure Layer 2 Tunnel Protocol Version 3" for more information on the CLI commands for L2TPv3.

Figure 2 L2TPv3 Header Format



Session ID

The L2TPv3 session ID is similar to the UTI session ID, and identifies the session context on the decapsulating system. For dynamic sessions, the value of the session ID is selected to optimize the context identification efficiency of the decapsulating system. A decapsulation implementation may therefore elect to support a smaller session ID bit field. In this L2TPv3 implementation, an upper value for the L2TPv3 session ID was set at 023. The L2TPv3 session ID value 0 is reserved for use by the protocol. For static sessions, the session ID is manually configured.

Note The local session ID must be unique on the decapsulating system and is restricted to the least significant ten bits.

Session Cookie

The L2TPv3 header contains a control channel cookie field that is similar to the UTI control channel key field. The control channel cookie field, however, has a variable length of 0, 4, or 8 bytes according to the cookie length supported by a given platform for packet decapsulation. The control channel cookie length can be manually configured for static sessions, or dynamically determined for dynamic sessions.

The variable cookie length does not present a problem when the same platform is at both ends of an L2TPv3 control channel. However, when different platforms interoperate across an L2TPv3 control channel, both platforms need to encapsulate packets with a 4-byte cookie length.

Pseudowire Control Encapsulation

The L2TPv3 pseudowire control encapsulation consists of 32 bits (4 bytes) and contains information used to sequence L2TP packets (see the section "Sequencing"). For the purposes of sequencing, only the first bit and bits 8 to 31 are relevant.

Bit 1 indicates whether the Sequence Number field, bits 8 to 31, contains a valid sequence number and is to be updated.

L2TPv3 Features

L2TPv3 provides Xconnect support for Ethernet, 802.1q (VLAN), Frame Relay, HDLC, and PPP, using the sessions described in the following sections:

- Static L2TPv3 Sessions (nonnegotiated, PVC-like forwarded sessions)
- Dynamic L2TPv3 Sessions (negotiated, forwarded sessions using the L2TPv3 control plane for session negotiation)

L2TPv3 also includes support for the features described in the following sections:

- Sequencing
- Local Switching
- Distributed Switching
- L2TPv3 Type of Service Marking
- Keepalive
- MTU Handling

Static L2TPv3 Sessions

Typically, the L2TP control plane is responsible for negotiating session parameters, such as the session ID or the cookie, in order to set up the session. However, some IP networks require sessions to be configured so that no signaling is required for session establishment. You can, therefore, set up static L2TPv3 sessions for a PE router by configuring fixed values for the fields in the L2TP data header. A static L2TPv3 session allows the PE to tunnel Layer 2 traffic as soon as the attachment circuit to which the session is bound comes up.

Note In an L2TPv3 static session, you can still run the L2TP control channel to perform peer authentication and dead-peer detection. If the L2TP control channel cannot be established or is torn down because of a hello failure, the static session is also torn down.

When you use a static L2TPv3 session, you cannot perform circuit interworking, such as LMI, because there is no facility to exchange control messages. To perform circuit interworking, you must use a dynamic session.

Dynamic L2TPv3 Sessions

A dynamic L2TP session is established through the exchange of control messages containing attribute-value pairs (AVPs). Each AVP contains information about the nature of the Layer 2 link being forwarded: the payload type, virtual circuit (VC) ID, and so on.

Multiple L2TP sessions (one for each forwarded Layer 2 circuit) can exist between a pair of PEs, and can be maintained by a single control channel. Session IDs and cookies are dynamically generated and exchanged as part of a dynamic session setup. Information such as sequencing configuration is also exchanged. Circuit state changes (UP/DOWN) are conveyed using the SLI message.

Sequencing

Although the correct sequence of received Layer 2 frames is guaranteed by some Layer 2 technologies (by the nature of the link, such as a serial line) or the protocol itself, forwarded Layer 2 frames may be lost, duplicated, or reordered when they traverse a network as IP packets. If the Layer 2 protocol does not provide an explicit sequencing mechanism, you can configure L2TP to sequence its data packets according to the data channel sequencing mechanism described in the L2TPv3 IETF l2tpext working group draft.

A receiver of L2TP data packets mandates sequencing through the Sequencing Required AVP when the session is being negotiated. A sender that receives this AVP (or that is manually configured to send sequenced packets) uses the Layer 2-specific pseudowire control encapsulation defined in L2TPv3.

You can configure L2TP only to drop out-of-order packets; you cannot configure L2TP to deliver the packets out-of-order. No reordering mechanism is available.

Local Switching

Local switching (from one port to another port in the same router) is supported for both static and dynamic sessions. You must configure separate IP addresses for each Xconnect statement.

See the section "Configuration Examples for Layer 2 Tunnel Protocol Version 3" for an example of how to configure local port switching.

Distributed Switching

dCEF switching is not supported for L2TP on the Cisco 7500 series Internet routers.

Note For the Cisco 7500 series, sequencing is supported, but all L2TP packets that require sequence number processing are sent to the RSP.

L2TPv3 Type of Service Marking

When Layer 2 traffic is tunneled across an IP network, information contained in the ToS bits may be transferred to the L2TP-encapsulated IP packets in one of the following ways:

- If the tunneled Layer 2 frames encapsulate IP packets themselves, it may be desirable to simply copy the ToS bytes of the inner IP packets to the outer IP packet headers. This action is known as "ToS byte reflection."
- Static ToS byte configuration. You specify the ToS byte value used by all packets sent across the pseudowire.

See the section "Configuring a Negotiated L2TPv3 Session for Local HDLC Switching Example" for more information about how to configure ToS information.

Keepalive

The keepalive mechanism for L2TPv3 extends only to the endpoints of the tunneling protocol. L2TP has a reliable control message delivery mechanism that serves as the basis for the keepalive mechanism. The keepalive mechanism consists of an exchange of L2TP hello messages.

If a keepalive mechanism is required, the control plane is used, although it may not be used to bring up sessions. You can manually configure sessions.

In the case of static L2TPv3 sessions, a control channel between the two L2TP peers is negotiated through the exchange of start control channel request (SCCRQ), start control channel replay (SCCRP), and start control channel connected (SCCCN) control messages. The control channel is responsible only for maintaining the keepalive mechanism through the exchange of hello messages.

The interval between hello messages is configurable per control channel. If one peer detects that the other has gone down through the keepalive mechanism, it sends a StopCCN control message and then notifies all of the pseudowires to the peer about the event. This notification results in the teardown of both manually configured and dynamic sessions.

MTU Handling

It is important that you configure a maximum transmission unit (MTU) appropriate for a each L2TPv3 tunneled link. The configured MTU size ensures the following:

- The lengths of the tunneled Layer 2 frames fall below the MTU of the destination attachment circuit
- The tunneled packets are not fragmented, which forces the receiving PE to reassemble them

L2TPv3 handles the MTU as follows:

- The default behavior is to fragment packets that are larger than the session MTU.
- If you enable the **ip dfbit set** command in the pseudowire class, the default MTU behavior changes so that any packets that cannot fit within the tunnel MTU are dropped.
- If you enable the **ip pmtu** command in the pseudowire class, the L2TPv3 control channel participates in the path MTU discovery. When you enable this feature, the following processing is performed:
  - Internet Control Message Protocol (ICMP) unreachable messages sent back to the L2TPv3 router are deciphered and the tunnel MTU is updated accordingly. In order to receive ICMP unreachable messages for fragmentation errors, the Don’t Fragment (DF) bit in the tunnel header is set according to the DF bit value received from the customer edge (CE) router, or statically if the **ip dfbit set** option is enabled. The tunnel MTU is periodically reset to the default value based on a periodic timer.
  - ICMP unreachable messages are sent back to the clients on the CE side. ICMP unreachable messages are sent to the CE whenever IP packets arrive on the CE-PE interface and have a packet size greater than the tunnel MTU. A Layer 2 header calculation is performed before the ICMP unreachable message is sent to the CE.

Supported L2TPv3 Payloads

L2TPv3 supports the following Layer 2 payloads that can be included in L2TPv3 packets tunneled over the pseudowire:

- Frame Relay
- Ethernet
- 802.1q (VLAN)
- HDLC
- PPP

Note Each L2TPv3 tunneled packet includes the entire Layer 2 frame of the payloads described in this section. If sequencing is required (see the section "Sequencing"), a Layer 2-specific sublayer (see the section "Pseudowire Control Encapsulation") is included in the L2TPv3 header to provide the Sequence Number field.

Frame Relay

L2TPv3 supports the Frame Relay functionality described in the following sections:

- Port-to-Port Trunking
- DLCI-to-DLCI Switching
- PVC Status Signaling
- Sequencing
- ToS Marking
- CIR Guarantees

Port-to-Port Trunking

Port-to-port trunking is where two CE Frame Relay interfaces are connected as by a leased line (UTI "raw" mode). All traffic arriving on one interface is forwarded transparently across the pseudowire to the other interface.

For example, in Figure 1, if the two CE routers are connected by a virtual leased line, the PE routers transparently transport all packets between CE R3 and CE R4 over a pseudowire. PE R1 and PE R2 do not examine or change the DLCIs, and do not participate in the LMI protocol. The two CE routers are LMI peers. There is nothing Frame Relay-specific about this service as far as the PE routers are concerned. The CE routers should be able to use any encapsulation based on HDLC framing without needing to change the provider configuration.

DLCI-to-DLCI Switching

Frame Relay DLCI-to-DLCI switching is where individual Frame Relay DLCIs are connected to create an end-to-end Frame Relay permanent virtual circuit (PVC). Traffic arriving on a DLCI on one interface is forwarded across the pseudowire to another DLCI on the other interface.

For example, in Figure 1, CE R3 and PE R1 are Frame Relay LMI peers; CE R4 and PE R2 are also LMI peers. You can use a different type of LMI between CE R3 and PE R1 compared to what you use between CE R4 and PE R2.

The CE devices may be a Frame Relay switch or end-user device. Each Frame Relay PVC is composed of multiple segments. The DLCI value is local to each segment and is changed as traffic is switched from segment to segment. Note that, in Figure 1, two Frame Relay PVC segments are connected by a pseudowire. Frame Relay header flags (FECN, BECN, C/R, DE) are preserved across the pseudowire.

PVC Status Signaling

PVC status signaling is propagated toward Frame Relay end users by the LMI protocol. You can configure the LMI to operate in any of the following modes:

- UNI DTE mode—PVC status is not reported, only received.
- UNI DCE mode—PVC status is reported but not received.
- NNI mode—PVC status is reported and received independently.

L2TPv3 supports all three modes.

The PVC status should be reported as ACTIVE only if the PVC is available from the reporting device to the Frame Relay end-user device. All interfaces, line protocols, and pseudowires must be operational between the reporting device and the Frame Relay end-user device.

Note that any keepalive functions on the session are independent of Frame Relay, but any state changes that are detected are fed into the PVC status reporting. For example, the L2TP control channel uses hello packets as a keepalive function. If the L2TPv3 keepalive fails, all L2TPv3 sessions are torn down. Loss of the session is notified to Frame Relay, which can then report PVCs INACTIVE to the CE devices.

For example, in Figure 1, CE R3 reports ACTIVE to PE R1 only if the PVC is available within CE R3. When CE R3 is a switch, it reports all the way to the user device in the customer network.

PE R1 reports ACTIVE to CE R3 only if the PVC is available within PE R1 and all the way to the end-user device (via PE R2 and CE R3) in the other customer VPN site.

The ACTIVE state is propagated hop-by-hop, independently in each direction, from one end of the Frame Relay network to the other end.

Sequencing

Frame Relay provides an ordered service in which packets sent to the Frame Relay network by one end-user device are delivered in order to the other end-user device. When switching is occurring over the pseudowire, packet ordering must be able to be preserved with a very high probability to closely emulate a traditional Frame Relay service. If the CE router is not using a protocol that can detect misordering itself, configuring sequence number processing may be important. For example, if the Layer 3 protocol is IP and Frame Relay is therefore used only for encapsulation, sequencing is not required. To detect misordering, you can configure sequence number processing separately for transmission or reception. For more information about how to configure sequencing, see the section "Configuring a Negotiated L2TPv3 Session for Local HDLC Switching Example."

ToS Marking

The ToS bytes in the IP header can be statically configured or reflected from the internal IP header. The Frame Relay DE bit does not influence the ToS bytes.

CIR Guarantees

In order to provide committed information rate (CIR) guarantees, you can configure a queueing policy that provides bandwidth to each DLCI to the interface facing the customer network on the egress PE.

Ethernet

An Ethernet frame arriving at a PE router is simply encapsulated in its entirety with an L2TP data header. At the other end, a received L2TP data packet is stripped of its L2TP data header. The payload, an Ethernet frame, is then forwarded to the appropriate attachment circuit.

Because the L2TPv3 tunneling protocol serves essentially as a bridge, it need not examine any part of an Ethernet frame. Any Ethernet frame received on an interface is tunneled, and any L2TP-tunneled Ethernet frame is forwarded out the interface.

Note Due to the way in which L2TPv3 handles Ethernet frames, an Ethernet interface must be configured to promiscuous mode in order to capture all traffic received on the Ethernet segment attached to the router. All frames will be tunneled through the L2TP pseudowire.

802.1q (VLAN)

L2TPv3 supports VLAN membership in the following ways:

- Port-based, in which undated Ethernet frames are received
- VLAN-based, in which tagged Ethernet frames are received

In L2TPv3, Ethernet Xconnect supports port-based VLAN membership and the reception of tagged Ethernet frames. A tagged Ethernet frame contains a tag header (defined in 802.1Q), which is 4 bytes long and consists of a 2-byte tag protocol identifier (TPID) field and a 2-byte tag control information (TCI) field. The TPID indicates that a TCI follows. The TCI is further broken down into the following three fields:

- User priority field
- Canonical format indicator (CFI)
- A 12-bit VLAN ID (VID)

For L2TPv3, an Ethernet subinterface configured to support VLAN switching may be bound to an Xconnect service so that all Ethernet traffic, tagged with a VID specified on the subinterface, is tunneled to another PE. The VLAN Ethernet frames are forwarded in their entirety. The receiving PE may rewrite the VID of the tunneled traffic to another value before forwarding the traffic onto an attachment circuit.

Note Due to the way in which L2TPv3 handles 802.1q VLAN packets, the Ethernet interface must be configured in promiscuous mode to capture all traffic received on the Ethernet segment attached to the router. All frames are tunneled through the L2TP pseudowire.

HDLC

L2TPv3 encapsulates an HDLC frame arriving at a PE in its entirety (including the Address, Control, and Protocol fields, but not the Flag fields and the frame check sequence) with an L2TP data header.

PPP

PEs that support L2TPv3 forward PPP traffic using a “transparent pass-through” model, in which the PEs play no role in the negotiation and maintenance of the PPP link. L2TPv3 encapsulates a PPP frame arriving at a PE in its entirety (including the HDLC Address and Control fields) with an L2TP data header.

How to Configure Layer 2 Tunnel Protocol Version 3

This section contains the following procedures:

- Configuring L2TP Control Channel Parameters, page 15 (optional)
- Configuring the L2TPv3 Pseudowire, page 19 (required)
- Configuring the Xconnect Attachment Circuit, page 22 (required)
- Manually Configuring L2TPv3 Session Parameters, page 24 (required)

Configuring L2TP Control Channel Parameters

The L2TP class configuration procedure creates a template of L2TP control channel parameters that can be inherited by different pseudowire classes. L2TP control channel parameters are used in control channel authentication, keepalive messages, and control channel negotiation. In an L2TPv3 session, the same L2TP class must be specified in the pseudowire configured on the PE router at each end of the control channel. Configuring L2TP control channel parameters is optional. However, the L2TP class must be configured before it is with associated a pseudowire class (see the section "Configuring the L2TPv3 Pseudowire").

The three main groups of L2TP control channel parameters that you can configure in an L2TP class are described in the following sections:

- Configuring L2TP Control Channel Timing Parameters
- Configuring L2TP Control Channel Authentication Parameters
- Configuring L2TP Control Channel Maintenance Parameters

After you enter L2TP class configuration mode, you can configure L2TP control channel parameters in any order. If you have multiple authentication requirements you can configure multiple sets of L2TP class control channel parameters with different L2TP class names. However, only one set of L2TP class control channel parameters can be applied to a connection between any pair of IP addresses.

Configuring L2TP Control Channel Timing Parameters

The following L2TP control channel timing parameters can be configured in L2TP class configuration mode:

- Packet size of the receive window used for the control channel
- Retransmission parameters used for control messages
- Timeout parameters used for the control channel

This task configures a set of timing control channel parameters in an L2TP class. All of the timing control channel parameter configurations are optional and may be configured in any order. If these parameters are not configured, the default values are applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2tp-class** [*l2tp-class-name*]
4. **receive-window** *size*

5. **retransmit** {**initial retries** *initial-retries* | **retries** *retries* | **timeout** {**max** | **min**} *timeout*}
6. **timeout setup** *seconds*

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.
Step 3	**l2tp-class** [*l2tp-class-name*] **Example:** `Router(config)# l2tp-class class1`	Specifies the L2TP class name and enters L2TP class configuration mode. • The *l2tp-class-name* argument is optional. However, if you want to configure multiple L2TP classes you must specify a unique *l2tp-class-name* for each one.
Step 4	**receive-window** *size* **Example:** `Router(config-l2tp-class)# receive-window 30`	(Optional) Configures the number of packets that can be received by the remote peer before backoff queueing occurs. • The valid values range from 1 to the upper limit the peer has for receiving packets. The default value is the upper limit.
Step 5	**retransmit** {**initial retries** *initial-retries* | **retries** *retries* | **timeout** {**max** | **min**} *timeout*} **Example:** `Router(config-l2tp-class)# retransmit retries 10`	(Optional) Configures parameters that affect the retransmission of control packets. • **initial retries**—specifies how many SCCRQs are re-sent before giving up on the session. Valid values for the *initial-retries* argument range from 1 to 1000. The default value is 2. • **retries**—specifies how many retransmission cycles occur before determining that the peer PE router does not respond. Valid values for the *retries* argument range from 1 to 1000. The default value is 15. • **timeout** {**max** | **min**}—specifies maximum and minimum retransmission intervals (in seconds) for resending control packets. Valid values for the *timeout* argument range from 1 to 8. The default maximum interval is 8; the default minimum interval is 1.
Step 6	**timeout setup** *seconds* **Example:** `Router(config-l2tp-class)# timeout setup 400`	(Optional) Configures the amount of time, in seconds, allowed to set up a control channel. • Valid values for the *seconds* argument range from 60 to 6000. The default value is 300.

Configuring L2TP Control Channel Authentication Parameters

The following L2TP control channel authentication parameters can be configured in L2TP class configuration mode:

- Authentication for the L2TP control channel
- Local host name used for authenticating the control channel
- Hiding the AVPs in outgoing control messages
- Password used for control channel authentication and AVP hiding

This task configures a set of authentication control channel parameters in an L2TP class. All of the authentication control channel parameter configurations are optional and may be configured in any order. If these parameters are not configured, the default values will be applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2tp-class** [*l2tp-class-name*]
4. **authentication**
5. **hostname** *name*
6. **hidden**
7. **password** [*encryption-type*] *password*

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.
Step 3	**l2tp-class** [*l2tp-class-name*] **Example:** `Router(config)# l2tp-class class1`	Specifies the L2TP class name and enters L2TP class configuration mode. • The *l2tp-class-name* argument is optional. However, if you want to configure multiple L2TP classes you must specify a unique *l2tp-class-name* for each one.
Step 4	**authentication** **Example:** `Router(config-l2tp-class)# authentication`	(Optional) Enables authentication for the control channel between PE routers.

	Command or Action	Purpose
Step 5	**hostname** *name* **Example:** `Router(config-l2tp-class)# hostname yb2`	(Optional) Specifies a host name used to identify the router during L2TP control channel authentication. • If you do not use this command, the default host name of the router is used.
Step 6	**hidden** **Example:** `Router(config-l2tp-class)# hidden`	(Optional) Hides the AVPs in control messages. • AVP hiding is not hidden by default.
Step 7	**password** [*encryption-type*] *password* **Example:** `Router(config-l2tp-class)# password tunnel2`	(Optional) Configures the password used for control channel authentication. • The valid values for the optional encryption type range from 0 to 7. If you do not use this command to specify a password, the password associated with the remote peer PE is taken from the value entered with the **username password** *value* global configuration command.

Configuring L2TP Control Channel Maintenance Parameters

The L2TP hello packet keepalive interval control channel maintenance parameter can be configured in L2TP class configuration mode.

This task configures the interval used for hello messages in an L2TP class. This control channel parameter configuration is optional. If this parameter is not configured, the default value will be applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2tp-class** [*l2tp-class-name*]
4. **hello** *interval*

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	**l2tp-class** [*l2tp-class-name*] **Example:** `Router(config)# l2tp-class class1`	Specifies the L2TP class name and enters L2TP class configuration mode. • The *l2tp-class-name* argument is optional. However, if you want to configure multiple L2TP classes you must specify a unique *l2tp-class-name* for each one.
Step 4	**hello** *interval* **Example:** `Router(config-l2tp-class)# hello 100`	(Optional) Specifies the exchange interval (in seconds) used between L2TP hello packets. • Valid values for the *interval* argument range from 0 to 1000. The default value is 60.

Configuring the L2TPv3 Pseudowire

The pseudowire class configuration procedure creates a configuration template for the pseudowire. You use this template, or class, to configure session-level parameters for L2TPv3 sessions that will be used to transport attachment circuit traffic over the pseudowire.

The pseudowire configuration specifies the characteristics of the L2TPv3 signaling mechanism, including the data encapsulation type, the control protocol, sequencing, fragmentation, payload-specific options, and IP properties. The setting that determines if signaling is used to set up the pseudowire is also included.

For simple L2TPv3 signaling configurations on most platforms, pseudowire class configuration is optional. However, specifying a source IP address to configure a loopback interface is highly recommended. If you do not configure a loopback interface, the router will choose the best available local address, which could be any IP address configured on a core-facing interface. This configuration could prevent a control channel from being established. If you do not configure other pseudowire class configuration commands, the default values are used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation l2tpv3**
5. **protocol** {**l2tpv3** | **none**} [*l2tp-class-name*]
6. **ip local interface** *interface-name*
7. **ip pmtu**
8. **ip tos** {**value** *value* | **reflect**}
9. **ip dfbit set**
10. **ip ttl** *value*
11. **ip protocol** {**l2tp** | **uti** | *protocol-number*}
12. **sequencing** {**transmit** | **receive** | **both**}

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.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config)# pseudowire-class etherpw`	Enters pseudowire class configuration mode and optionally specifies the name of the L2TP pseudowire class.
Step 4	**encapsulation l2tpv3** **Example:** `Router(config-pw)# encapsulation l2tpv3`	Specifies that L2TPv3 is used as the data encapsulation method to tunnel IP traffic.
Step 5	**protocol** {**l2tpv3** | **none**}[*l2tp-class-name*] **Example:** `Router(config-pw)# protocol l2tpv3 class1`	(Optional) Specifies the L2TPv3 signaling protocol to be used to manage the pseudowires created with the control channel parameters in the specified L2TP class (see the section "Configuring L2TP Control Channel Parameters"). • If the *l2tp-class-name* argument is not specified, the default values for L2TP control channel parameters will be used. The default **protocol** option is **l2tpv3**. • If you do not want to use signaling in the L2TPv3 sessions created with this pseudowire class, enter **protocol none**. (The **protocol none** configuration is necessary when configuring interoperability with a remote peer that runs UTI.)
Step 6	**ip local interface** *interface-name* **Example:** `Router(config-pw)# ip local interface e0/0`	Specifies the PE router interface whose IP address is to be used as the source IP address for sending tunneled packets. • Use the same local interface name for all pseudowire classes configured between a pair of PE routers. **Note** This command must be configured for pseudowire-class configurations using L2TPv3 as the data encapsulation method.

	Command or Action	Purpose
Step 7	**ip pmtu** **Example:** `Router(config-pw)# ip pmtu`	(Optional) Enables the discovery of the path MTU for tunneled traffic. • This command enables the processing of ICMP unreachable messages that indicate fragmentation errors in the backbone network that carries L2TPv3 session traffic. Also, this command enables MTU checking for IP packets sent into the session and that have the DF bit set. Any IP packet larger than the MTU is dropped and an ICMP unreachable message is sent. MTU discovery is disabled by default. **Note** The **ip pmtu** command is not supported if you disabled signaling with the **protocol none** command in Step 5. • This command must be enabled in the pseudowire class configuration for fragmentation of IP packets before the data enters the pseudowire to occur. **Note** For fragmentation of IP packets before the data enters the pseudowire, it is recommended that the **ip dfbit set** command is also enabled in the pseudowire class configuration. This allows the PMTU to be obtained more rapidly.
Step 8	**ip tos** {**value** *value* | **reflect**} **Example:** `Router(config-pw)# ip tos reflect`	(Optional) Configures the value of the ToS byte in IP headers of tunneled packets, or reflects the ToS byte value from the inner IP header. • Valid values for the *value* argument range from 0 to 255. The default ToS byte value is 0.
Step 9	**ip dfbit set** **Example:** `Router(config-pw)# ip dfbit set`	(Optional) Configures the value of the DF bit in the outer headers of tunneled packets. • Use this command if (for performance reasons) you do not want reassembly of tunneled packets to be performed on the peer PE router. This command is disabled by default.
Step 10	**ip ttl** *value* **Example:** `Router(config-pw)# ip ttl 100`	(Optional) Configures the value of the time to live (TTL) byte in the IP headers of tunneled packets. • Valid values for the *value* argument range from 1 to 255. The default TTL byte value is 255.

	Command or Action	Purpose
Step 11	**ip protocol** {**l2tp** | **uti** | *protocol-number*} **Example:** `Router(config-pw)# ip protocol uti`	(Optional) Configures the IP protocol to be used for tunneling packets. • For backward compatibility with UTI, enter **uti** or **120**, the UTI protocol number. The default IP protocol value is **l2tp** or **115**, the L2TP protocol number.
Step 12	**sequencing** {**transmit** | **receive** | **both**} **Example:** `Router(config-pw)# sequencing both`	(Optional) Specifies the direction in which sequencing of data packets in a pseudowire is enabled: • **transmit**—Updates the Sequence Number field in the headers of data packets sent over the pseudowire according to the data encapsulation method that is used. • **receive**—Keeps the Sequence Number field in the headers of data packets received over the pseudowire. Out-of-order packets are dropped. • **both**—Enables both the **transmit** and **receive** options.

Configuring the Xconnect Attachment Circuit

This configuration procedure binds an Ethernet, 802.1q VLAN, or Frame Relay attachment circuit to an L2TPv3 pseudowire for Xconnect service. The virtual circuit identifier that you configure creates the binding between a pseudowire configured on a PE router and an attachment circuit in a CE device. The virtual circuit identifier configured on the PE router at one end of the L2TPv3 control channel must also be configured on the peer PE router at the other end.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot/port*
4. **xconnect** *peer-ip-address vcid pseudowire-parameters* [**sequencing** {**transmit** | **receive** | **both**}]

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	**interface** *type slot/port* **Example:** `Router(config)# interface ethernet 0/0`	Specifies the interface by type (for example, Ethernet) and slot and port number, and enters interface configuration mode.
Step 4	**xconnect** *peer-ip-address vcid pseudowire-parameters* [**sequencing** {**transmit** | **receive** | **both**}] **Example:** `Router(config-if)# xconnect 10.0.3.201 123 pw-class vlan-xconnect`	Specifies the IP address of the peer PE router and the 32-bit virtual circuit identifier shared between the PE at each end of the control channel. The peer router IP address and virtual circuit ID must be a unique combination on the router. At least one of the following pseudowire class parameters must be configured for the *pseudowire-parameters* argument: • **encapsulation** {**l2tpv3** [**manual**] | **mpls**}—Specifies the tunneling method used to encapsulate data in the pseudowire: – **l2tpv3**—L2TPv3 is the tunneling method to be used. – **manual**—(Optional) No signaling is to be used in the L2TPv3 control channel. This command places the router in xconnect configuration mode for manual configuration of L2TPv3 parameters for the attachment circuit. – **mpls**—MPLS is the tunneling method to be used. • **pw-class** *pw-class-name*—The pseudowire class configuration from which the data encapsulation type (L2TPv3) will be taken. The optional **encapsulation** parameter specifies the method of pseudowire tunneling used: L2TPv3 or MPLS. Enter **manual** if you do not want signaling used in the L2TPv3 control channel. The **encapsulation l2tpv3 manual** keyword combination enters xconnect configuration submode. See the section "Manually Configuring L2TPv3 Session Parameters" for the other L2TPv3 commands that you must enter to complete the configuration of the L2TPv3 control channel. If you do not enter an **encapsulation** value, the encapsulation method entered with the **password** command in the section "Configuring L2TP Control Channel Authentication Parameters" is used. The optional **pw-class** parameter binds the Xconnect statement to a specific pseudowire class. The pseudowire class then serves as the template configuration for all attachment circuits bound to it. Specify the pseudowire-class option if you need to configure more advanced options. **Note** You must configure either the **encapsulation** or the **pw-class** option. You may configure both options. **Note** If you select L2TPv3 as your data encapsulation method, you must specify the **pw-class** keyword. The optional **sequencing** parameter specifies whether sequencing is required for packets that are received, sent, or both received and sent.

Manually Configuring L2TPv3 Session Parameters

When you bind an attachment circuit to an L2TPv3 pseudowire for Xconnect service using the **xconnect l2tpv3 manual** command (see the section "Configuring the Xconnect Attachment Circuit") because you do not want signaling, you must then configure L2TP-specific parameters to complete the L2TPv3 control channel configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot/port*
4. **xconnect** *peer-ip-address vc-id* **encapsulation l2tpv3 manual pw-class** *pw-class-name*
5. **l2tp id** *local-session-id remote-session-id*
6. **l2tp cookie local** *size low-value* [*high-value*]
7. **l2tp cookie remote** *size low-value* [*high-value*]
8. **l2tp hello** *l2tp-class-name*

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.
Step 3	**interface** *type slot/port* **Example:** `Router(config)# interface ethernet 0/0`	Specifies the interface by type (for example, Ethernet) and slot and port number, and enters interface configuration mode.
Step 4	**xconnect** *peer-ip-address vc-id* **encapsulation l2tpv3 manual pw-class** *pw-class-name* **Example:** `Router(config-if)# xconnect 10.0.3.201 123 encapsulation l2tpv3 manual pw-class vlan-xconnect`	Specifies the IP address of the peer PE router and the 32-bit virtual circuit identifier shared between the PE at each end of the control channel. • The peer router ID (IP address) and virtual circuit ID must be a unique combination on the router. • The **encapsulation l2tpv3 manual** parameter specifies that L2TPv3 is to be used as the pseudowire tunneling method, and enters xconnect configuration mode. • The mandatory **pw-class** *pw-class-name* keyword and argument combination specifies the pseudowire class configuration from which the data encapsulation type (L2TPv3) will be taken.

	Command or Action	Purpose
Step 5	**l2tp id** *local-session-id remote-session-id* **Example:** `Router(config-if-xconn)# l2tp id 222 111`	Configures the identifiers for the local L2TPv3 session and for the remote L2TPv3 session on the peer PE router. • This command is required to complete the attachment circuit configuration and for a static L2TPv3 session configuration.
Step 6	**l2tp cookie local** *size low-value* [*high-value*] **Example:** `Router(config-l2tp-class)# l2tp cookie local 4 54321`	(Optional) Specifies the value that the peer PE must include in the cookie field of incoming (received) L2TP packets. • The size of the cookie field can be 4 or 8 bytes. If you do not enter this command, no cookie value is included in the header of L2TP packets. • If you configure the cookie length in incoming packets as 8 bytes, you must specify a 4-byte high value and a 4-byte low value.
Step 7	**l2tp cookie remote** *size low-value* [*high-value*] **Example:** `Router(config-l2tp-class)# l2tp cookie remote 4 12345`	(Optional) Specifies the value that the router includes in the cookie field of outgoing (sent) L2TP packets. • The size of the cookie field can be 4 or 8 bytes. If you do not enter this command, no cookie value is included in the header of L2TP packets. • If you configure the cookie length in outgoing packets as 8 bytes, you must specify a 4-byte high value and a 4-byte low value.
Step 8	**l2tp hello** *l2tp-class-name* **Example:** `Router(config-l2tp-class)# l2tp hello l2tp-defaults`	(Optional) Specifies the L2TP class name to use (see the section "Configuring L2TP Control Channel Parameters") for control channel configuration parameters, including the interval to use between hello keepalive messages. **Note** This command assumes that there is no control plane to negotiate control channel parameters and that a control channel is to be used to provide keepalive support through an exchange of L2TP hello messages. By default, no hello messages are sent.

Configuration Examples for Layer 2 Tunnel Protocol Version 3

This section provides the following configuration examples:

Configuring Frame Relay DLCI-to-DLCI Switching Example

The following is a sample configuration for switching a Frame Relay DLCI over a pseudowire:

```
pseudowire-class fr-xconnect
 encapsulation l2tpv3
 protocol l2tpv3
 ip local interface Loopback0
 sequencing both

interface Serial0/0
 encapsulation frame-relay
 frame-relay intf-type dce

connect one Serial0/0 100 l2transport
 xconnect 10.0.3.201 555 pw-class fr-xconnect

connect two Serial0/0 200 l2transport
 xconnect 10.0.3.201 666 pw-class fr-xconnect
```

Configuring Frame Relay Trunking Example

The following is a sample configuration for setting up a trunk connection for an entire serial interface over a pseudowire. All incoming packets are switched to the pseudowire regardless of content.

Note that when you configure trunking for a serial interface, the trunk connection does not require an encapsulation method. You do not, therefore, need to enter the **encapsulation frame-relay** command. Reconfiguring the default encapsulation removes all Xconnect configuration settings from the interface.

```
interface Serial0/0
 xconnect 10.0.3.201 555 pw-class serial-xconnect
```

Configuring an MQC for Committed Information Rate Guarantees Example

The following is a sample configuration of the MQC to guarantee a CIR of 256 kbps on DLCI 100 and 512 kbps for DLCI 200:

```
ip cef distributed
 class-map dlci100
 match fr-dlci 100
 class-map dlci200
 match fr-dlci 200

policy-map dlci
 class dlci100
 bandwidth 256
 class dlci200
 bandwidth 512

interface Serial0/0
 encapsulation frame-relay
 frame-relay intf-type dce
 service-policy output dlci
```

```
connect one Serial0/0 100 l2transport
 xconnect 10.0.3.201 555 encapsulation l2tpv3 pw-class mqc

connect two Serial0/0 200 l2transport
 xconnect 10.0.3.201 666 encapsulation l2tpv3 pw-class mqc
```

Configuring a Static L2TPv3 Session for an Xconnect Ethernet Interface Example

L2TPv3 is the only encapsulation method that supports a manually provisioned session setup. This example shows how to configure a static session configuration in which all control channel parameters are set up in advance. There is no control plane used and no negotiation phase to set up the control channel. The PE router starts sending tunneled traffic as soon as the Ethernet interface (int e0/0) comes up. The virtual circuit identifier, 123, is not used. The PE sends L2TP data packets with session ID 111 and cookie 12345. In turn, the PE expects to receive L2TP data packets with session ID 222 and cookie 54321.

```
l2tp-class l2tp-defaults
 retransmit initial retries 30
 cookie-size 8

pseudowire-class ether-pw
 encapsulation l2tpv3
 protocol none
 ip local interface Loopback0

interface Ethernet 0/0
 xconnect 10.0.3.201 123 encapsulation l2tpv3 manual pw-class ether-pw
 l2tp id 222 111
 l2tp cookie local 4 54321
 l2tp cookie remote 4 12345
 l2tp hello l2tp-defaults
```

Configuring a Negotiated L2TPv3 Session for an Xconnect VLAN Subinterface Example

The following is a sample configuration of a dynamic L2TPv3 session for a VLAN Xconnect interface. In this example, only VLAN traffic with a VLAN ID of 5 is tunneled. In the other direction, the L2TPv3 session identified by a virtual circuit identifier of 123 receives forwarded frames whose VLAN ID fields are rewritten to contain the value 5. L2TPv3 is used as both the control plane protocol and the data encapsulation.

```
l2tp-class class1
 authentication
 password secret

pseudowire-class vlan-xconnect
 encapsulation l2tpv3
 protocol l2tpv3 class1
 ip local interface Loopback0

interface Ethernet0/0.1
 encapsulation dot1Q 5
 xconnect 10.0.3.201 123 pw-class vlan-xconnect
```

Configuring a Negotiated L2TPv3 Session for Local HDLC Switching Example

The following is a sample configuration of a dynamic L2TPv3 session for local HDLC switching. In this example, note that it is necessary to configure two different IP addresses at the endpoints of the L2TPv3 pseudowire because the virtual circuit identifier must be unique for a given IP address.

```
interface loopback 1
 ip address 10.0.0.1 255.255.255.255

interface loopback 2
 ip address 10.0.0.2 255.255.255.255

pseudowire-class loopback1
 encapsulation l2tpv3
 ip local interface loopback1

pseudowire-class loopback2
 encapsulation l2tpv3
 ip local interface loopback2

interface s0/0
 encapsulation hdlc
 xconnect 10.0.0.1 100 pw-class loopback2

interface s0/1
 encapsulation hdlc
 xconnect 10.0.0.2 100 pw-class loopback1
```

Verifying an L2TPv3 Session Example

To display detailed information about current L2TPv3 sessions on a router, use the **show l2tun session all** command:

```
Router# show l2tunnel session all

Session Information Total tunnels 0 sessions 1

Session id 111 is up, tunnel id 0
Call serial number is 0
Remote tunnel name is
  Internet address is 2.0.0.1
  Session is manually signalled
  Session state is established, time since change 00:06:05
    0 Packets sent, 0 received
    0 Bytes sent, 0 received
    Receive packets dropped:
      out-of-order:             0
      total:                    0
    Send packets dropped:
      exceeded session MTU:     0
      total:                    0
  Session vcid is 123
  Session Layer 2 circuit, type is ATM VPC CELL, name is ATM3/0/0:1000007
  Circuit state is UP
    Remote session id is 222, remote tunnel id 0
  DF bit off, ToS reflect disabled, ToS value 0, TTL value 255
  Session cookie information:
    local cookie, size 8 bytes, value 00 00 00 00 00 00 00 64
    remote cookie, size 8 bytes, value 00 00 00 00 00 00 00 C8
  SSS switching enabled
  Sequencing is off
```

Verifying an L2TP Control Channel Example

To display detailed information the L2TP control channels that are set up to other L2TP-enabled devices for all L2TP sessions on the router, use the **show l2tun tunnel all** command. The L2TP control channel is used to negotiate capabilities, monitor the health of the peer PE router, and set up various components of an L2TPv3 session.

```
Router# show l2tun session all

Session Information Total tunnels 0 sessions 1

Session id 111 is up, tunnel id 0
Call serial number is 0
Remote tunnel name is
  Internet address is 2.0.0.1
  Session is manually signalled
  Session state is established, time since change 00:06:05
    0 Packets sent, 0 received
    0 Bytes sent, 0 received
    Receive packets dropped:
      out-of-order:             0
      total:                    0
    Send packets dropped:
      exceeded session MTU:     0
      total:                    0
  Session vcid is 123
  Session Layer 2 circuit, type is ATM VPC CELL, name is ATM3/0/0:1000007
  Circuit state is UP
    Remote session id is 222, remote tunnel id 0
  DF bit off, ToS reflect disabled, ToS value 0, TTL value 255
  Session cookie information:
    local cookie, size 8 bytes, value 00 00 00 00 00 00 00 64
    remote cookie, size 8 bytes, value 00 00 00 00 00 00 00 C8
  SSS switching enabled
  Sequencing is off
```

Additional References

The following sections provide additional information related to the L2TPv3 feature.

Related Documents

Related Topic	Document Title
Further information about L2TPv3	*Layer 2 Tunneling Protocol Version 3 Technical Overview*
Information about L2TP	*Layer 2 Tunnel Protocol* *Layer 2 Tunneling Protocol: A Feature in Cisco IOS Software*
Configuring the CEF feature	"Cisco Express Forwarding" chapter in the *Cisco IOS Switching Services Configuration Guide*
Further information about MTU discovery and packet fragmentation	*MTU Tuning for L2TP*

Related Topic	Document Title
Additional VPN commands: complete command syntax, command mode, defaults, usage guidelines and examples	*Cisco IOS Technologies Command Reference, Release 12.3*
Additional Frame Relay commands: complete command syntax, command mode, defaults, usage guidelines and examples	*Cisco IOS Wide-Area Networking Command Reference, Release 12.3*
Information about UTI	*Universal Transport Interface (UTI)*

Standards

Standards	Title
draft-ietf-l2tpext-l2tp-base-03.txt	*Layer Two Tunneling Protocol (Version 3)'L2TPv3'*

MIBs

MIBs	MIBs Link
VPDN MIB—The Cisco VPDN Management MIB (CISCO-VPDN-MGMT-MIB.my) provides objects for the monitoring of L2TPv3 activity using SNMP.	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 2661	*Layer Two Tunneling Protocol "L2TP"*

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, tools, and lots more. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

This section documents new and modified commands. All other commands used with this feature are documented in the Cisco IOS Release 12.3 command reference publications.

- **authentication**
- **debug acircuit**
- **debug xconnect**
- **encapsulation l2tpv3**
- **hello**
- **hidden**
- **hostname**
- **ip dfbit set**
- **ip local interface**
- **ip pmtu**
- **ip protocol**
- **ip tos**
- **ip ttl**
- **l2tp-class**
- **l2tp cookie local**
- **l2tp cookie remote**
- **l2tp hello**
- **l2tp id**
- **password**
- **protocol**
- **pseudowire-class**
- **receive-window**
- **retransmit**
- **sequencing**
- **show l2tun session**
- **show l2tun tunnel**
- **snmp-server enable traps l2tun session**
- **timeout setup**
- **xconnect**

Glossary

AV pairs—attribute-value pairs.

BECN—backward explicit congestion notification. Bit set by a Frame Relay network in frames traveling in the opposite direction of frames encountering a congested path. DTE receiving frames with the BECN bit set can request that higher-level protocols take flow control action as appropriate.

CE—customer edge (Frame Relay switch or user device).

CIR—committed information rate. Rate at which a Frame Relay network agrees to transfer information under normal conditions, averaged over a minimum increment of time. CIR, measured in bits per second, is one of the key negotiated tariff metrics.

data-link control layer—Layer 2 in the SNA architectural model. Responsible for the transmission of data over a particular physical link. Corresponds approximately to the data link layer of the OSI model.

DCE—data circuit-terminating equipment (ITU-T expansion). Devices and connections of a communications network that comprise the network end of the user-to-network interface.

dCEF—distributed Cisco Express Forwarding.

DLCI—data-link connection identifier. A unique number assigned to a PVC endpoint in a Frame Relay network. Identifies a particular PVC endpoint within an access channel in a Frame Relay network and has local significance only to that channel.

DTE—data terminal equipment. Device at the user end of a user-network interface that serves as a data source, destination, or both.

FECN—forward explicit congestion notification. Bit set by a Frame Relay network to inform DTE receiving the frame that congestion was experienced in the path from source to destination. DTE receiving frames with the FECN bit set can request that higher-level protocols take flow-control action as appropriate.

HDLC—High-Level Data Link Control. A generic link-level communications protocol developed by the International Organization for Standardization (ISO). HDLC manages synchronous, code-transparent, serial information transfer over a link connection.

ICMP—Internet Control Message Protocol. A network protocol that handles network errors and error messages.

IDB—interface descriptor block.

IS-IS—Intermediate System-to-Intermediate System. OSI link-state hierarchical routing protocol based on DECnet Phase V routing, whereby ISs (routers) exchange routing information based on a single metric to determine network topology.

L2TP—An extension to PPP merging features of two tunneling protocols: Layer 2 Forwarding (L2F) from Cisco Systems and Point-to-Point Tunneling (PPTP) from Microsoft. L2TP is an Internet Engineering Task Force (IETF) standard endorsed by Cisco Systems, and other networking industry leaders.

L2TPv3—Draft version of L2TP that enhances functionality in RFC 2661 (L2TP).

LMI—Local Management Interface.

MPLS—Multiprotocol Label Switching. Switching method that forwards IP traffic using a label. This label instructs the routers and the switches in the network where to forward the packets based on preestablished IP routing information.

MQC—modular quality of service command-line interface.

MTU—maximum transmission unit. Maximum packet size, in bytes, that a particular interface can handle.

NNI—Network-to-Network Interface. ATM Forum standard that defines the interface between two ATM switches that are both located in a private network or are both located in a public network. The UNI standard defines the interface between a public switch and a private one. Also, the standard interface between two Frame Relay switches meeting the same criteria.

PE—Provider edge router providing Frame Relay over L2TPv3 functionality.

PPP—Point-to-Point Protocol. A link-layer encapsulation method for dialup or dedicated circuits. A successor to Serial Line IP (SLIP), PPP provides router-to-router and host-to-network connections over synchronous and asynchronous circuits.

PVC—permanent virtual circuit. A virtual circuit that is permanently established. A Frame Relay logical link, whose endpoints and class of service are defined by network management. Analogous to an X.25 permanent virtual circuit, a PVC consists of the originating Frame Relay network element address, originating data-link control identifier, terminating Frame Relay network element address, and termination data-link control identifier. Originating refers to the access interface from which the PVC is initiated. Terminating refers to the access interface at which the PVC stops. Many data network customers require a PVC between two points. PVCs save bandwidth associated with circuit establishment and tear down in situations where certain virtual circuits must exist all the time. Data terminating equipment with a need for continuous communication uses PVCs.

PW—pseudowire.

SNMP—Simple Network Management Protocol. Network management protocol used almost exclusively in TCP/IP networks. SNMP provides a means to monitor and control network devices, and to manage configurations, statistics collection, performance, and security.

tunneling—Architecture that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme.

UNI—User-Network Interface.

UTI—Universal Transport Interface.

VPDN—virtual private dialup network. A network that allows separate and autonomous protocol domains to share common access infrastructure, including modems, access servers, and ISDN routers. A VPDN enables users to configure secure networks that take advantage of ISPs that tunnel remote access traffic through the ISP cloud.

Note Refer to *Internetworking Terms and Acronyms* for terms not included in this glossary.