Cisco IOS Bridging and IBM Networking Configuration Guide

Release 12.4

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Cisco IOS Bridging and IBM Networking Configuration Guide

About Cisco IOS and Cisco IOS XE Software Documentation

Last updated: August 6, 2008

This document describes the objectives, audience, conventions, and organization used in Cisco IOS and Cisco IOS XE software documentation, collectively referred to in this document as Cisco IOS documentation. Also included are resources for obtaining technical assistance, additional documentation, and other information from Cisco. This document is organized into the following sections:

Documentation Objectives

Cisco IOS documentation describes the tasks and commands available to configure and maintain Cisco networking devices.

Audience

The Cisco IOS documentation set is i ntended for users who configure and maintain Cisco networking devices (such as routers and switches) but who may not be familiar with the configuration and maintenance tasks, the relationship among tasks, or the Cisco IOS commands necessary to perform particular tasks. The Cisco IOS documentation set is also intended for those users experienced with Cisco IOS who need to know about new features, new configuration options, and new software characteristics in the current Cisco IOS release.

Documentation Conventions

In Cisco IOS documentation, the term *router* may be used to refer to various Cisco products; for example, routers, access servers, and switches. These and other networking devices that support Cisco IOS software are shown interchangeably in examples and are used only for illustrative purposes. An example that shows one product does not necessarily mean that other products are not supported.

This section includes the following topics:

Typographic Conventions

Cisco IOS documentation uses the following typographic conventions:

Convention	Description
^ or Ctrl	Both the ^ symbol and Ctrl represent the Control (Ctrl) key on a keyboard. For example, the key combination **^D** or **Ctrl-D** means that you hold down the Control key while you press the D key. (Keys are indicated in capital letters but are not case sensitive.)
string	A string is a nonquoted set of characters shown in italics. For example, when setting a Simple Network Management Protocol (SNMP) community string to *public*, do not use quotation marks around the string; otherwise, the string will include the quotation marks.

Command Syntax Conventions

Cisco IOS documentation uses the following command syntax conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter as shown.
italic	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional keyword or argument.
|	A vertical line, called a pipe, indicates a choice within a set of keywords or arguments.
[x | y]	Square brackets enclosing keywords or arguments separated by a pipe indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a pipe indicate a required choice.
[x {y | z}]	Braces and a pipe within square brackets indicate a required choice within an optional element.

Software Conventions

Cisco IOS uses the following program code conventions:

Convention	Description
`Courier font`	Courier font is used for information that is displayed on a PC or terminal screen.
`Bold Courier font`	Bold Courier font indicates text that the user must enter.
< >	Angle brackets enclose text that is not displayed, such as a password. Angle brackets also are used in contexts in which the italic font style is not supported; for example, ASCII text.
!	An exclamation point at the beginning of a line indicates that the text that follows is a comment, not a line of code. An exclamation point is also displayed by Cisco IOS software for certain processes.
[ ]	Square brackets enclose default responses to system prompts.

Reader Alert Conventions

The Cisco IOS documentation set uses the following conventions for reader alerts:

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Documentation Organization

This section describes the Cisco IOS documentation set, how it is organized, and how to access it on Cisco.com. Included are lists of configuration guides, command references, and supplementary references and resources that make up the documentation set. The following topics are included:

Cisco IOS Documentation Set

Cisco IOS documentation consists of the following:

- Release notes and caveats provide information about platform, technology, and feature support for a release and describe severity 1 (catastrophic), severity 2 (severe), and severity 3 (moderate) defects in released Cisco IOS code. Review release notes before other documents to learn whether or not updates have been made to a feature.
- Sets of configuration guides and command references organized by technology and published for each standard Cisco IOS release.
  - Configuration guides—Compilations of documents that provide informational and task-oriented descriptions of Cisco IOS features.
  - Command references—Compilations of command pages that provide detailed information about the commands used in the Cisco IOS features and processes that make up the related configuration guides. For each technology, there is a single command reference that covers all Cisco IOS releases and that is updated at each standard release.
- Lists of all the commands in a specific release and all commands that are new, modified, removed, or replaced in the release.
- Command reference book for **debug** commands. Command pages are listed in alphabetical order.
- Reference book for system messages for all Cisco IOS releases.

Cisco IOS Documentation on Cisco.com

The following sections describe the documentation organization and how to access various document types.

Use Cisco Feature Navigator to find information about platform support and Cisco IOS and Catalyst OS software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

New Features List

The New Features List for each release provides a list of all features in the release with hyperlinks to the feature guides in which they are documented.

Feature Guides

Cisco IOS features are documented in feature guides. Feature guides describe one feature or a group of related features that are supported on many different software releases and platforms. Your Cisco IOS software release or platform may not support all the features documented in a feature guide. See the Feature Information table at the end of the feature guide for information about which features in that guide are supported in your software release.

Configuration Guides

Configuration guides are provided by technology and release and comprise a set of individual feature guides relevant to the release and technology.

Command References

Command reference books describe Cisco IOS commands that are supported in many different software releases and on many different platforms. The books are provided by technology. For information about all Cisco IOS commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or the *Cisco IOS Master Command List, All Releases*, at http://www.cisco.com/en/US/docs/ios/mcl/all_release/all_mcl.html.

Cisco IOS Supplementary Documents and Resources

Supplementary documents and resources are listed in Table 2 on page xi.

Configuration Guides, Command References, and Supplementary Resources

Table 1 lists, in alphabetical order, Cisco IOS and Cisco IOS XE software configuration guides and command references, including brief descriptions of the contents of the documents. The Cisco IOS command references are comprehensive, meaning that they include commands for both Cisco IOS software and Cisco IOS XE software, for all releases. The configuration guides and command references support many different software releases and platforms. Your Cisco IOS software release or platform may not support all these technologies.

For additional information about configuring and operating specific networking devices, go to the Product Support area of Cisco.com at http://www.cisco.com/web/psa/products/index.html.

Table 2 lists documents and resources that supplement the Cisco IOS software configuration guides and command references. These supplementary resources include release notes and caveats; master command lists; new, modified, removed, and replaced command lists; system messages; and the debug command reference.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS AppleTalk Configuration Guide *Cisco IOS XE AppleTalk Configuration Guide* *Cisco IOS AppleTalk Command Reference*	AppleTalk protocol.
Cisco IOS Asynchronous Transfer Mode Configuration Guide *Cisco IOS Asynchronous Transfer Mode Command Reference*	LAN ATM, multiprotocol over ATM (MPoA), and WAN ATM.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Bridging and IBM Networking Configuration Guide *Cisco IOS Bridging Command Reference* *Cisco IOS IBM Networking Command Reference*	• Transparent and source-route transparent (SRT) bridging, source-route bridging (SRB), Token Ring Inter-Switch Link (TRISL), and token ring route switch module (TRRSM). • Data-link switching plus (DLSw+), serial tunnel (STUN), block serial tunnel (BSTUN); logical link control, type 2 (LLC2), synchronous data link control (SDLC); IBM Network Media Translation, including Synchronous Data Logical Link Control (SDLLC) and qualified LLC (QLLC); downstream physical unit (DSPU), Systems Network Architecture (SNA) service point, SNA frame relay access, advanced peer-to-peer networking (APPN), native client interface architecture (NCIA) client/server topologies, and IBM Channel Attach.
Cisco IOS Broadband and DSL Configuration Guide *Cisco IOS XE Broadband and DSL Configuration Guide* *Cisco IOS Broadband and DSL Command Reference*	Point-to-Point Protocol (PPP) over ATM (PPPoA) and PPP over Ethernet (PPPoE).
Cisco IOS Carrier Ethernet Configuration Guide *Cisco IOS Carrier Ethernet Command Reference*	Connectivity fault management (CFM), Ethernet Local Management Interface (ELMI), IEEE 802.3ad link bundling, Link Layer Discovery Protocol (LLDP), media endpoint discovery (MED), and operations, administration, and maintenance (OAM).
Cisco IOS Configuration Fundamentals Configuration Guide *Cisco IOS XE Configuration Fundamentals Configuration Guide* *Cisco IOS Configuration Fundamentals Command Reference*	Autoinstall, Setup, Cisco IOS command-line interface (CLI), Cisco IOS file system (IFS), Cisco IOS web browser user interface (UI), basic file transfer services, and file management.
Cisco IOS DECnet Configuration Guide *Cisco IOS XE DECnet Configuration Guide* *Cisco IOS DECnet Command Reference*	DECnet protocol.
Cisco IOS Dial Technologies Configuration Guide *Cisco IOS XE Dial Technologies Configuration Guide* *Cisco IOS Dial Technologies Command Reference*	Asynchronous communications, dial backup, dialer technology, dial-in terminal services and AppleTalk remote access (ARA), large scale dialout, dial-on-demand routing, dialout, modem and resource pooling, ISDN, multilink PPP (MLP), PPP, virtual private dialup network (VPDN).
Cisco IOS Flexible NetFlow Configuration Guide *Cisco IOS Flexible NetFlow Command Reference*	Flexible NetFlow.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS H.323 Configuration Guide	Gatekeeper enhancements for managed voice services, Gatekeeper Transaction Message Protocol, gateway codec order preservation and shutdown control, H.323 dual tone multifrequency relay, H.323 version 2 enhancements, Network Address Translation (NAT) support of H.323 v2 Registration, Admission, and Status (RAS) protocol, tokenless call authorization, and VoIP gateway trunk and carrier-based routing.
Cisco IOS High Availability Configuration Guide *Cisco IOS XE High Availability Configuration Guide* *Cisco IOS High Availability Command Reference*	A variety of High Availability (HA) features and technologies that are available for different network segments (from enterprise access to service provider core) to facilitate creation of end-to-end highly available networks. Cisco IOS HA features and technologies can be categorized in three key areas: system-level resiliency, network-level resiliency, and embedded management for resiliency.
Cisco IOS Integrated Session Border Controller Command Reference	A VoIP-enabled device that is deployed at the edge of networks. An SBC is a toolkit of functions, such as signaling interworking, network hiding, security, and quality of service (QoS).
Cisco IOS Intelligent Service Gateway Configuration Guide *Cisco IOS Intelligent Service Gateway Command Reference*	Subscriber identification, service and policy determination, session creation, session policy enforcement, session life-cycle management, accounting for access and service usage, session state monitoring.
Cisco IOS Interface and Hardware Component Configuration Guide *Cisco IOS XE Interface and Hardware Component Configuration Guide* *Cisco IOS Interface and Hardware Component Command Reference*	LAN interfaces, logical interfaces, serial interfaces, virtual interfaces, and interface configuration.
Cisco IOS IP Addressing Services Configuration Guide *Cisco IOS XE Addressing Services Configuration Guide* *Cisco IOS IP Addressing Services Command Reference*	Address Resolution Protocol (ARP), Network Address Translation (NAT), Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Next Hop Address Resolution Protocol (NHRP).
Cisco IOS IP Application Services Configuration Guide *Cisco IOS XE IP Application Services Configuration Guide* *Cisco IOS IP Application Services Command Reference*	Enhanced Object Tracking (EOT), Gateway Load Balancing Protocol (GLBP), Hot Standby Router Protocol (HSRP), IP Services, Server Load Balancing (SLB), Stream Control Transmission Protocol (SCTP), TCP, Web Cache Communication Protocol (WCCP), User Datagram Protocol (UDP), and Virtual Router Redundancy Protocol (VRRP).
Cisco IOS IP Mobility Configuration Guide *Cisco IOS IP Mobility Command Reference*	Mobile ad hoc networks (MANet) and Cisco mobile networks.
Cisco IOS IP Multicast Configuration Guide *Cisco IOS XE IP Multicast Configuration Guide* *Cisco IOS IP Multicast Command Reference*	Protocol Independent Multicast (PIM) sparse mode (PIM-SM), bidirectional PIM (bidir-PIM), Source Specific Multicast (SSM), Multicast Source Discovery Protocol (MSDP), Internet Group Management Protocol (IGMP), and Multicast VPN (MVPN).

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS IP Routing Protocols Configuration Guide *Cisco IOS XE IP Routing Protocols Configuration Guide* *Cisco IOS IP Routing Protocols Command Reference*	Border Gateway Protocol (BGP), multiprotocol BGP, multiprotocol BGP extensions for IP multicast, bidirectional forwarding detection (BFD), Enhanced Interior Gateway Routing Protocol (EIGRP), Interior Gateway Routing Protocol (IGRP), Intermediate System-to-Intermediate System (IS-IS), on-demand routing (ODR), Open Shortest Path First (OSPF), and Routing Information Protocol (RIP).
Cisco IOS IP SLAs Configuration Guide *Cisco IOS XE IP SLAs Configuration Guide* *Cisco IOS IP SLAs Command Reference*	Cisco IOS IP Service Level Agreements (IP SLAs).
Cisco IOS IP Switching Configuration Guide *Cisco IOS XE IP Switching Configuration Guide* *Cisco IOS IP Switching Command Reference*	Cisco Express Forwarding, fast switching, and Multicast Distributed Switching (MDS).
Cisco IOS IPv6 Configuration Guide *Cisco IOS XE IPv6 Configuration Guide* *Cisco IOS IPv6 Command Reference*	For IPv6 features, protocols, and technologies, go to the IPv6 "Start Here" document at the following URL: http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-roadmap.html
Cisco IOS ISO CLNS Configuration Guide *Cisco IOS XE ISO CLNS Configuration Guide* *Cisco IOS ISO CLNS Command Reference*	ISO connectionless network service (CLNS).
Cisco IOS LAN Switching Configuration Guide *Cisco IOS XE LAN Switching Configuration Guide* *Cisco IOS LAN Switching Command Reference*	VLANs, Inter-Switch Link (ISL) encapsulation, IEEE 802.10 encapsulation, IEEE 802.1Q encapsulation, and multilayer switching (MLS).
Cisco IOS Mobile Wireless Gateway GPRS Support Node Configuration Guide *Cisco IOS Mobile Wireless Gateway GPRS Support Node Command Reference*	Cisco IOS Gateway GPRS Support Node (GGSN) in a 2.5-generation general packet radio service (GPRS) and 3-generation universal mobile telecommunication system (UMTS) network.
Cisco IOS Mobile Wireless Home Agent Configuration Guide *Cisco IOS Mobile Wireless Home Agent Command Reference*	Cisco Mobile Wireless Home Agent, an anchor point for mobile terminals for which mobile IP or proxy mobile IP services are provided.
Cisco IOS Mobile Wireless Packet Data Serving Node Configuration Guide *Cisco IOS Mobile Wireless Packet Data Serving Node Command Reference*	Cisco Packet Data Serving Node (PDSN), a wireless gateway that is between the mobile infrastructure and standard IP networks and that enables packet data services in a code division multiple access (CDMA) environment.
Cisco IOS Mobile Wireless Radio Access Networking Configuration Guide *Cisco IOS Mobile Wireless Radio Access Networking Command Reference*	Cisco IOS radio access network products.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Multiprotocol Label Switching Configuration Guide *Cisco IOS XE Multiprotocol Label Switching Configuration Guide* *Cisco IOS Multiprotocol Label Switching Command Reference*	MPLS Label Distribution Protocol (LDP), MPLS Layer 2 VPNs, MPLS Layer 3 VPNs, MPLS Traffic Engineering (TE), and MPLS Embedded Management (EM) and MIBs.
Cisco IOS Multi-Topology Routing Configuration Guide *Cisco IOS Multi-Topology Routing Command Reference*	Unicast and multicast topology configurations, traffic classification, routing protocol support, and network management support.
Cisco IOS NetFlow Configuration Guide *Cisco IOS XE NetFlow Configuration Guide* *Cisco IOS NetFlow Command Reference*	Network traffic data analysis, aggregation caches, export features.
Cisco IOS Network Management Configuration Guide *Cisco IOS XE Network Management Configuration Guide* *Cisco IOS Network Management Command Reference*	Basic system management; system monitoring and logging; troubleshooting, logging, and fault management; Cisco Discovery Protocol; Cisco IOS Scripting with Tool Control Language (Tcl); Cisco networking services (CNS); DistributedDirector; Embedded Event Manager (EEM); Embedded Resource Manager (ERM); Embedded Syslog Manager (ESM); HTTP; Remote Monitoring (RMON); SNMP; and VPN Device Manager Client for Cisco IOS Software (XSM Configuration).
Cisco IOS Novell IPX Configuration Guide *Cisco IOS XE Novell IPX Configuration Guide* *Cisco IOS Novell IPX Command Reference*	Novell Internetwork Packet Exchange (IPX) protocol.
Cisco IOS Optimized Edge Routing Configuration Guide *Cisco IOS Optimized Edge Routing Command Reference*	Optimized edge routing (OER) monitoring, policy configuration, routing control, logging and reporting, and VPN IPsec/generic routing encapsulation (GRE) tunnel interface optimization.
Cisco IOS Quality of Service Solutions Configuration Guide *Cisco IOS XE Quality of Service Solutions Configuration Guide* *Cisco IOS Quality of Service Solutions Command Reference*	Class-based weighted fair queuing (CBWFQ), custom queuing, distributed traffic shaping (DTS), generic traffic shaping (GTS), IP- to-ATM class of service (CoS), low latency queuing (LLQ), modular QoS CLI (MQC), Network-Based Application Recognition (NBAR), priority queuing, Security Device Manager (SDM), Multilink PPP (MLPPP) for QoS, header compression, AutoQoS, QoS features for voice, Resource Reservation Protocol (RSVP), weighted fair queuing (WFQ), and weighted random early detection (WRED).
Cisco IOS Security Configuration Guide *Cisco IOS XE Security Configuration Guide* *Cisco IOS Security Command Reference*	Access control lists (ACLs), authentication, authorization, and accounting (AAA), firewalls, IP security and encryption, neighbor router authentication, network access security, network data encryption with router authentication, public key infrastructure (PKI), RADIUS, TACACS+, terminal access security, and traffic filters.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Service Selection Gateway Configuration Guide *Cisco IOS Service Selection Gateway Command Reference*	Subscriber authentication, service access, and accounting.
Cisco IOS Software Activation Configuration Guide *Cisco IOS Software Activation Command Reference*	An orchestrated collection of processes and components to activate Cisco IOS software feature sets by obtaining and validating Cisco software licenses.
Cisco IOS Software Modularity Installation and Configuration Guide *Cisco IOS Software Modularity Command Reference*	Installation and basic configuration of software modularity images, including installations on single and dual route processors, installation rollbacks, software modularity binding, software modularity processes and patches.
Cisco IOS Terminal Services Configuration Guide *Cisco IOS Terminal Services Command Reference* *Cisco IOS XE Terminal Services Command Reference*	DEC, local-area transport (LAT), and X.25 packet assembler/disassembler (PAD).
Cisco IOS Virtual Switch Command Reference	Virtual switch redundancy, high availability, and packet handling; converting between standalone and virtual switch modes; virtual switch link (VSL); Virtual Switch Link Protocol (VSLP). **Note** For information about virtual switch configuration, refer to the product-specific software configuration information for the Cisco Catalyst 6500 series switch or for the Metro Ethernet 6500 series switch.
Cisco IOS Voice Configuration Library *Cisco IOS Voice Command Reference*	Cisco IOS support for voice call control protocols, interoperability, physical and virtual interface management, and troubleshooting. The library includes documentation for IP telephony applications.
Cisco IOS VPDN Configuration Guide *Cisco IOS XE VPDN Configuration Guide* *Cisco IOS VPDN Command Reference*	Layer 2 Tunneling Protocol (L2TP) dial-out load balancing and redundancy, L2TP extended failover, L2TP security VPDN, multihop by Dialed Number Identification Service (DNIS), timer and retry enhancements for L2TP and Layer 2 Forwarding (L2F), RADIUS Attribute 82: tunnel assignment ID, shell-based authentication of VPDN users, tunnel authentication via RADIUS on tunnel terminator.
Cisco IOS Wide-Area Networking Configuration Guide *Cisco IOS XE Wide-Area Networking Configuration Guide* *Cisco IOS Wide-Area Networking Command Reference*	Frame Relay, Layer 2 Tunneling Protocol Version 3 (L2TPv3), Link Access Procedure, Balanced (LAPB), Switched Multimegabit Data Service (SMDS), and X.25.
Cisco IOS Wireless LAN Configuration Guide *Cisco IOS Wireless LAN Command Reference*	Broadcast key rotation, IEEE 802.11x support, IEEE 802.1x authenticator, IEEE 802.1x local authentication service for Extensible Authentication Protocol-Flexible Authentication via Secure Tunneling (EAP-FAST), Multiple Basic Service Set ID (BSSID), Wi-Fi Multimedia (WMM) required elements, and Wi-Fi Protected Access (WPA).

Table 2 Cisco IOS Supplementary Documents and Resources

Document Title	Description
Cisco IOS Master Command List, All Releases	Alphabetical list of all the commands documented in all Cisco IOS releases.
Cisco IOS New, Modified, Removed, and Replaced Commands	List of all the new, modified, removed, and replaced commands for a Cisco IOS release.
Cisco IOS Software System Messages	List of Cisco IOS system messages and descriptions. System messages may indicate problems with your system; be informational only; or may help diagnose problems with communications lines, internal hardware, or the system software.
Cisco IOS Debug Command Reference	Alphabetical list of **debug** commands including brief descriptions of use, command syntax, and usage guidelines.
Release Notes and Caveats	Information about new and changed features, system requirements, and other useful information about specific software releases; information about defects in specific Cisco IOS software releases.
MIBs	Files used for network monitoring. To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator at the following URL: http://www.cisco.com/go/mibs
RFCs	Standards documents maintained by the Internet Engineering Task Force (IETF) that Cisco IOS documentation references where applicable. The full text of referenced RFCs may be obtained at the following URL: http://www.rfc-editor.org/

Additional Resources and Documentation Feedback

What's New in Cisco Product Documentation is published monthly and describes all new and revised Cisco technical documentation. The *What's New in Cisco Product Documentation* publication also provides information about obtaining the following resources:

- Technical documentation
- Cisco product security overview
- Product alerts and field notices
- Technical assistance

Cisco IOS technical documentation includes embedded feedback forms where you can rate documents and provide suggestions for improvement. Your feedback helps us improve our documentation.

Using the Command-Line Interface in Cisco IOS and Cisco IOS XE Software

Last updated: August 6, 2008

This document provides basic information about the command-line interface (CLI) in Cisco IOS and Cisco IOS XE software and how you can use some of the CLI features. This document contains the following sections:

For more information about using the CLI, see the "Using the Cisco IOS Command-Line Interface" section of the *Cisco IOS Configuration Fundamentals Configuration Guide*.

For information about the software documentation set, see the "About Cisco IOS and Cisco IOS XE Software Documentation" document.

Initially Configuring a Device

Initially configuring a device varies by platform. For information about performing an initial configuration, see the hardware installation documentation that is provided with the original packaging of the product or go to the Product Support area of Cisco.com at http://www.cisco.com/web/psa/products/index.html.

After you have performed the initial configuration and connected the device to your network, you can configure the device by using the console port or a remote access method, such as Telnet or Secure Shell (SSH), to access the CLI or by using the configuration method provided on the device, such as Security Device Manager.

Changing the Default Settings for a Console or AUX Port

There are only two changes that you can make to a console port and an AUX port:

- Change the port speed with the **config-register 0x** command. Changing the port speed is not recommended. The well-known default speed is 9600.
- Change the behavior of the port; for example, by adding a password or changing the timeout value.

Note The AUX port on the Route Processor (RP) installed in a Cisco ASR1000 series router does not serve any useful customer purpose and should be accessed only under the advisement of a customer support representative.

Using the CLI

This section describes the following topics:

Understanding Command Modes

The CLI command mode structure is hierarchical, and each mode supports a set of specific commands. This section describes the most common of the many modes that exist.

Table 1 lists common command modes with associated CLI prompts, access and exit methods, and a brief description of how each mode is used.

Table 1 CLI Command Modes

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
User EXEC	Log in.	`Router>`	Issue the **logout** or **exit** command.	• Change terminal settings. • Perform basic tests. • Display device status.
Privileged EXEC	From user EXEC mode, issue the **enable** command.	`Router#`	Issue the **disable** command or the **exit** command to return to user EXEC mode.	• Issue **show** and **debug** commands. • Copy images to the device. • Reload the device. • Manage device configuration files. • Manage device file systems.
Global configuration	From privileged EXEC mode, issue the **configure terminal** command.	`Router(config)#`	Issue the **exit** command or the **end** command to return to privileged EXEC mode.	Configure the device.
Interface configuration	From global configuration mode, issue the **interface** command.	`Router(config-if)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual interfaces.
Line configuration	From global configuration mode, issue the **line vty** or **line console** command.	`Router(config-line)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual terminal lines.

Table 1 CLI Command Modes (continued)

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
ROM monitor	From privileged EXEC mode, issue the **reload** command. Press the **Break** key during the first 60 seconds while the system is booting.	`rommon # >` The # symbol represents the line number and increments at each prompt.	Issue the **continue** command.	• Run as the default operating mode when a valid image cannot be loaded. • Access the fall-back procedure for loading an image when the device lacks a valid image and cannot be booted. • Perform password recovery when a CTRL-Break sequence is issued within 60 seconds of a power-on or reload event.
Diagnostic (available only on the Cisco ASR1000 series router)	The router boots or enters diagnostic mode in the following scenarios. When a Cisco IOS process or processes fail, in most scenarios the router will reload. • A user-configured access policy was configured using the **transport-map** command, which directed the user into diagnostic mode. • The router was accessed using an RP auxiliary port. • A break signal (**Ctrl-C**, **Ctrl-Shift-6**, or the **send break** command) was entered, and the router was configured to enter diagnostic mode when the break signal was received.	`Router(diag)#`	If a Cisco IOS process failure is the reason for entering diagnostic mode, the failure must be resolved and the router must be rebooted to exit diagnostic mode. If the router is in diagnostic mode because of a transport-map configuration, access the router through another port or using a method that is configured to connect to the Cisco IOS CLI. If the RP auxiliary port was used to access the router, use another port for access. Accessing the router through the auxiliary port is not useful for customer purposes.	• Inspect various states on the router, including the Cisco IOS state. • Replace or roll back the configuration. • Provide methods of restarting the Cisco IOS software or other processes. • Reboot hardware, such as the entire router, an RP, an ESP, a SIP, a SPA, or possibly other hardware components. • Transfer files into or off of the router using remote access methods such as FTP, TFTP, and SCP.

EXEC commands are not saved when the software reboots. Commands that you issue in a configuration mode can be saved to the startup configuration. If you save the running configuration to the startup configuration, these commands will execute when the software is rebooted. Global configuration mode is the highest level of configuration mode. From global configuration mode, you can enter a variety of other configuration modes, including protocol-specific modes.

ROM monitor mode is a separate mode that is used when the software cannot load properly. If a valid software image is not found when the software boots or if the configuration file is corrupted at startup, the software might enter ROM monitor mode. Use the question symbol (?) to view the commands that you can use while the device is in ROM monitor mode.

```
rommon 1 > ?
alias               set and display aliases command
boot                boot up an external process
confreg             configuration register utility
cont                continue executing a downloaded image
context             display the context of a loaded image
cookie              display contents of cookie PROM in hex
.
.
.
rommon 2 >
```

The following example shows how the command prompt changes to indicate a different command mode:

```
Router> enable
Router# configure terminal
Router(config)# interface ethernet 1/1
Router(config-if)# ethernet
Router(config-line)# exit
Router(config)# end
Router#
```

Note A keyboard alternative to the **end** command is Ctrl-Z.

Using the Interactive Help Feature

The CLI includes an interactive Help feature. Table 2 describes how to use the Help feature.

Table 2 CLI Interactive Help Commands

Command	Purpose
help	Provides a brief description of the help feature in any command mode.
?	Lists all commands available for a particular command mode.
*partial command***?**	Provides a list of commands that begin with the character string (no space between the command and the question mark).
partial command<**Tab**>	Completes a partial command name (no space between the command and <Tab>).
command **?**	Lists the keywords, arguments, or both associated with the command (space between the command and the question mark).
command keyword **?**	Lists the arguments that are associated with the keyword (space between the keyword and the question mark).

The following examples show how to use the help commands:

help

```
Router> help

Help may be requested at any point in a command by entering a question mark '?'.  If
nothing matches, the help list will be empty and you must backup until entering a '?'
shows the available options.

Two styles of help are provided:

1. Full help is available when you are ready to enter a command argument (e.g. 'show ?')
and describes each possible argument.

2. Partial help is provided when an abbreviated argument is entered and you want to know
what arguments match the input (e.g. 'show pr?'.)
```

?

```
Router# ?
Exec commands:
  access-enable        Create a temporary access-List entry
  access-profile       Apply user-profile to interface
  access-template      Create a temporary access-List entry
  alps                 ALPS exec commands
  archive              manage archive files
<snip>
```

***partial command*?**

```
Router(config)# zo?
zone  zone-pair
```

***partial command*<Tab>**

```
Router(config)# we<Tab> webvpn
```

***command* ?**

```
Router(config-if)# pppoe ?
  enable        Enable pppoe
  max-sessions  Maximum PPPOE sessions
```

***command keyword* ?**

```
Router(config-if)# pppoe enable ?
  group  attach a BBA group
  <cr>
```

Understanding Command Syntax

Command syntax is the format in which a command should be entered in the CLI. Commands include the name of the command, keywords, and arguments. Keywords are alphanumeric strings that are used literally. Arguments are placeholders for values that a user must supply. Keywords and arguments may be required or optional.

Specific conventions convey information about syntax and command elements. Table 3 describes these conventions.

Table 3 CLI Syntax Conventions

Symbol/Text	Function	Notes
< > (angle brackets)	Indicate that the option is an argument.	Sometimes arguments are displayed without angle brackets.
A.B.C.D.	Indicates that you must enter a dotted decimal IP address.	Angle brackets (< >) are not always used to indicate that an IP address is an argument.
WORD (all capital letters)	Indicates that you must enter one word.	Angle brackets (< >) are not always used to indicate that a WORD is an argument.
LINE (all capital letters)	Indicates that you must enter more than one word.	Angle brackets (< >) are not always used to indicate that a LINE is an argument.
<cr> (carriage return)	Indicates the end of the list of available keywords and arguments, and also indicates when keywords and arguments are optional. When <cr> is the only option, you have reached the end of the branch or the end of the command if the command has only one branch.	—

The following examples show syntax conventions:

```
Router(config)# ethernet cfm domain ?
  WORD  domain name
Router(config)# ethernet cfm domain dname ?
  level
Router(config)# ethernet cfm domain dname level ?
  <0-7>  maintenance level number
Router(config)# ethernet cfm domain dname level 7 ?
  <cr>
Router(config)# snmp-server file-transfer access-group 10 ?
  protocol  protocol options
  <cr>
Router(config)# logging host ?
  Hostname or A.B.C.D  IP address of the syslog server
  ipv6                 Configure IPv6 syslog server
Router(config)# snmp-server file-transfer access-group 10 ?
  protocol  protocol options
  <cr>
```

Understanding Enable and Enable Secret Passwords

Some privileged EXEC commands are used for actions that impact the system, and it is recommended that you set a password for these commands to prevent unauthorized use. Two types of passwords, enable (not encrypted) and enable secret (encrypted), can be set. The following commands set these passwords and are issued in global configuration mode:

- **enable** *password*
- **enable secret** *password*

Using an enable secret password is recommended because it is encrypted and more secure than the enable password. When you use an enable secret password, text is encrypted (unreadable) before it is written to the config.text file. When you use an enable password, the text is written as entered (readable) to the config.text file.

Each type of password is case sensitive, can contain from 1 to 25 uppercase and lowercase alphanumeric characters, and can start with a number. Spaces are also valid password characters; for example, "two words" is a valid password. Leading spaces are ignored, but trailing spaces are recognized.

Note Both password commands have numeric keywords that are single integer values. If you choose a number for the first character of your password followed by a space, the system will read the number as if it were the numeric keyword and not as part of your password.

When both passwords are set, the enable secret password takes precedence over the enable password.

To remove a password, use the **no** form of the commands: **no enable** *password* or **no enable secret** *password*.

For more information about password recovery procedures for Cisco products, see http://www.cisco.com/en/US/products/sw/iosswrel/ps1831/products_tech_note09186a00801746e6.shtml.

Using the Command History Feature

The CLI command history feature saves the commands you enter during a session in a command history buffer. The default number of commands saved is 10, but the number is configurable within the range of 0 to 256. This command history feature is particularly useful for recalling long or complex commands.

To change the number of commands saved in the history buffer for a terminal session, issue the **terminal history size** command:

```
Router# terminal history size num
```

A command history buffer is also available in line configuration mode with the same default and configuration options. To set the command history buffer size for a terminal session in line configuration mode, issue the **history** command:

```
Router(config-line)# history [size num]
```

To recall commands from the history buffer, use the following methods:

- Press Ctrl-P or the up arrow key—Recalls commands beginning with the most recent command. Repeat the key sequence to recall successively older commands.

- Press Ctrl-N or the down arrow key—Recalls the most recent commands in the history buffer after they have been recalled using Ctrl-P or the up arrow key. Repeat the key sequence to recall successively more recent commands.

Note The arrow keys function only on ANSI-compatible terminals such as the VT100.

- Issue the **show history** command in user EXEC or privileged EXEC mode—Lists the most recent commands that you entered. The number of commands that are displayed is determined by the setting of the **terminal history size** and **history** commands.

  The CLI command history feature is enabled by default. To disable this feature for a terminal session, issue the **terminal no history** command in user EXEC or privileged EXEC mode or the **no history** command in line configuration mode.

Abbreviating Commands

Typing a complete command name is not always required for the command to execute. The CLI recognizes an abbreviated command when the abbreviation contains enough characters to uniquely identify the command. For example, the **show version** command can be abbreviated as **sh ver**. It cannot be abbreviated as **s ver** because **s** could mean **show**, **set**, or **systat**. The **sh v** abbreviation also is not valid because the **show** command has **vrrp** as a keyword in addition to **version**. (Command and keyword examples from Cisco IOS Release 12.4(13)T.)

Using Aliases for CLI Commands

To save time and the repetition of entering the same command multiple times, you can use a command alias. An alias can be configured to do anything that can be done at the command line, but an alias cannot move between modes, type in passwords, or perform any interactive functions.

Table 4 shows the default command aliases.

Table 4 Default Command Aliases

Command Alias	Original Command
h	help
lo	logout
p	ping
s	show
u or **un**	undebug
w	where

To create a command alias, issue the **alias** command in global configuration mode. The syntax of the command is **alias** *mode command-alias original-command*. Following are some examples:

- Router(config)# **alias exec prt partition**—privileged EXEC mode
- Router(config)# **alias configure sb source-bridge**—global configuration mode
- Router(config)# **alias interface rl rate-limit**—interface configuration mode

To view both default and user-created aliases, issue the **show alias** command.

For more information about the **alias** command, see http://www.cisco.com/en/US/docs/ios/fundamentals/command/reference/cf_book.html.

Using the no and default Forms of Commands

Most configuration commands have a **no** form that is used to reset a command to its default value or disable a feature or function. For example, the **ip routing** command is enabled by default. To disable this command, you would issue the **no ip routing** command. To re-enable IP routing, you would issue the **ip routing** command.

Configuration commands may also have a **default** form, which returns the command settings to their default values. For commands that are disabled by default, using the **default** form has the same effect as using the **no** form of the command. For commands that are enabled by default and have default settings, the **default** form enables the command and returns the settings to their default values.

The **no** and **default** forms of commands are described in the command pages of command references.

Using the debug Command

A **debug** command produces extensive output that helps you troubleshoot problems in your network. These commands are available for many features and functions within Cisco IOS and Cisco IOS XE software. Some **debug** commands are **debug all**, **debug aaa accounting**, and **debug mpls packets**. To use **debug** commands during a Telnet session with a device, you must first enter the **terminal monitor** command. To turn off debugging completely, you must enter the **undebug all** command.

For more information about **debug** commands, see the *Cisco IOS Debug Command Reference* at http://www.cisco.com/en/US/docs/ios/debug/command/reference/db_book.html.

Caution Debugging is a high priority and high CPU utilization process that can render your device unusable. Use **debug** commands only to troubleshoot specific problems. The best times to run debugging are during periods of low network traffic and when few users are interacting with the network. Debugging during these periods decreases the likelihood that the **debug** command processing overhead will affect network performance or user access or response times.

Filtering Output Using Output Modifiers

Many commands produce lengthy output that may use several screens to display. Using output modifiers, you can filter this output to show only the information that you want to see.

Three output modifiers are available and are described as follows:

- **begin** *regular expression*—Displays the first line in which a match of the regular expression is found and all lines that follow.
- **include** *regular expression*—Displays all lines in which a match of the regular expression is found.
- **exclude** *regular expression*—Displays all lines except those in which a match of the regular expression is found.

To use one of these output modifiers, type the command followed by the pipe symbol (|), the modifier, and the regular expression that you want to search for or filter. A regular expression is a case-sensitive alphanumeric pattern. It can be a single character or number, a phrase, or a more complex string.

The following example illustrates how to filter output of the **show interface** command to display only lines that include the expression "protocol."

```
Router# show interface | include protocol

FastEthernet0/0 is up, line protocol is up
Serial4/0 is up, line protocol is up
Serial4/1 is up, line protocol is up
Serial4/2 is administratively down, line protocol is down
Serial4/3 is administratively down, line protocol is down
```

Understanding CLI Error Messages

You may encounter some error messages while using the CLI. Table 5 shows the common CLI error messages.

Table 5 Common CLI Error Messages

Error Message	Meaning	How to Get Help
% Ambiguous command: "show con"	You did not enter enough characters for the command to be recognized.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Incomplete command.	You did not enter all the keywords or values required by the command.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Invalid input detected at "^" marker.	You entered the command incorrectly. The caret (^) marks the point of the error.	Enter a question mark (?) to display all the commands that are available in this command mode. The keywords that you are allowed to enter for the command appear.

For more system error messages, see the following documents:

- *Cisco IOS Release 12.2SR System Message Guide*
- *Cisco IOS System Messages, Volume 1 of 2* (Cisco IOS Release 12.4)
- *Cisco IOS System Messages, Volume 2 of 2* (Cisco IOS Release 12.4)

Saving Changes to a Configuration

To save changes that you made to the configuration of a device, you must issue the **copy running-config startup-config** command or the **copy system:running-config nvram:startup-config** command. When you issue these commands, the configuration changes that you made are saved to the startup configuration and saved when the software reloads or power to the device is turned off or interrupted. The following example shows the syntax of the **copy running-config startup-config** command:

```
Router# copy running-config startup-config
Destination filename [startup-config]?
```

You press Enter to accept the startup-config filename (the default), or type a new filename and then press Enter to accept that name. The following output is displayed indicating that the configuration was saved:

```
Building configuration...
[OK]
Router#
```

On most platforms, the configuration is saved to NVRAM. On platforms with a Class A flash file system, the configuration is saved to the location specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM.

Additional Information

- "Using the Cisco IOS Command-Line Interface" section of the *Cisco IOS Configuration Fundamentals Configuration Guide*:

  http://www.cisco.com/en/US/docs/ios/fundamentals/configuration/guide/cf_cli-basics.html

  or

  "Using Cisco IOS XE Software" chapter of the *Cisco ASR1000 Series Aggregation Services Routers Software Configuration Guide*:

  http://www.cisco.com/en/US/docs/routers/asr1000/configuration/guide/chassis/using_cli.html
- Cisco Product Support Resources

  http://www.cisco.com/web/psa/products/index.html
- Support area on Cisco.com (also search for documentation by task or product)

  http://www.cisco.com/en/US/support/index.html
- *White Paper: Cisco IOS Reference Guide*

  http://www.cisco.com/en/US/products/sw/iosswrel/ps1828/products_white_paper09186a008018305e.shtml
- Software Download Center (downloads; tools; licensing, registration, advisory, and general information) (requires Cisco.com User ID and password)

  http://www.cisco.com/kobayashi/sw-center/
- Error Message Decoder, a tool to help you research and resolve error messages for Cisco IOS software

  http://www.cisco.com/pcgi-bin/Support/Errordecoder/index.cgi

- Command Lookup Tool, a tool to help you find detailed descriptions of Cisco IOS commands (requires Cisco.com user ID and password)

  http://tools.cisco.com/Support/CLILookup

- Output Interpreter, a troubleshooting tool that analyzes command output of supported **show** commands

  https://www.cisco.com/pcgi-bin/Support/OutputInterpreter/home.pl\

Overview of Cisco SNA Internetworking

The *Cisco IOS Bridging and IBM Networking Configuration Guide* discusses software components used to internetwork mainframe-based Systems Network Architecture (SNA) networks with router-based TCP/IP networks. This overview chapter provides a description of Cisco SNA internetworking. For technology overview and configuration information, refer to the appropriate chapter in this publication. This chapter contains the following sections:

Background and Overview

IBM mainframes and SNA traditionally have formed the foundation of enterprise networks. In the 1980s, Cisco routers and TCP/IP emerged as the technologies for the future of enterprise networks. By the early 1990s, many large commercial, government, and educational organizations began to integrate TCP/IP products and technologies into their SNA networks. Today, the common denominator for electronic communication from one organization to another or from a consumer to a company is TCP/IP. Adopting a TCP/IP infrastructure is the first logical step to creating a multiservice network that seamlessly accommodates data, voice, and video.

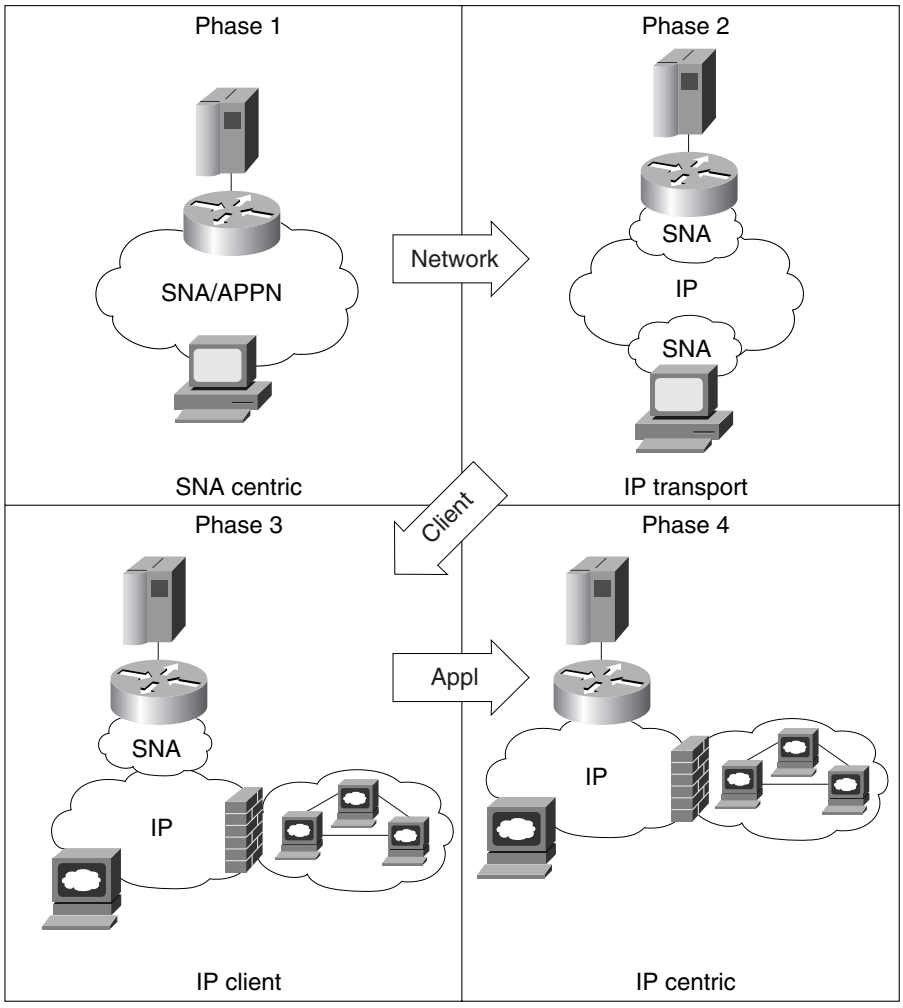
Enterprise organizations are heavily invested in mainframes and SNA and mainframes are still a vital part of enterprise data centers. The goal for these enterprise organizations is to integrate the TCP/IP-based environment with the SNA-based environment. The Cisco bridging and IBM networking technologies enable the delivery of SNA data over routers supporting TCP/IP.

The Cisco Four-Phase Model for SNA-to-IP Integration

Cisco has developed a high-level, four-phase model illustrating a typical integration path to incorporate TCP/IP into an SNA-based network. Figure 1 illustrates the four-phase integration path. The model helps to describe some common phases in SNA-to-IP integration. A single phase in this integration path might represent the network of some organizations, while two or more phases might represent the network implementation of other organizations in various sectors of their network.

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

Figure 1 The Cisco Four-Phase SNA-to-IP Integration Model



The phases can be differentiated by the protocol that runs in each of three key elements in the network: the mainframe/midrange computer, the network backbone, and the desktop. The characteristics of each of the phases are described here along with the problems solved, types of products and technologies implemented, and challenges.

This section contains the following topics:

Phase One: SNA Centric

An SNA-centric network has SNA, Advanced Peer-to-Peer Networking (APPN), or APPN/High Performance Routing (HPR) protocols running on one or more mainframe/midrange systems, in the network backbone, and at the desktop. Subarea networks that were widely implemented in the 1980s were built upon ACF/VTAM in the mainframe, ACF/Network Control Point (NCP) in communication processors (that is, front-end processors [FEPs] and remote concentrator processors [RCPs]), and cluster controllers with terminals attached via coaxial cable. The communication lines utilized were predominantly leased Synchronous Data Link Control (SDLC) and public or private X.25 lines.

In the late 1980s and early 1990s, traditional SNA networks evolved to meet the new demands for client/server computing and LANs. PCs running terminal emulation software replaced many of the fixed-function terminals. Token Ring LANs were widely deployed to bring higher speeds and support client/server computing. RCPs were often replaced by a new generation of remote SNA devices—LAN gateways, bridge/routers, and Frame Relay access devices (FRADs).

Today's SNA-centric network is a very high-speed and dynamic network when compared to the traditional SNA network of the past. ACF/VTAM on the mainframe includes APPN/HPR protocols to support dynamic rerouting around failures and high-speed switching in the network. The mainframe complex, which now comprises multiple complementary metal-oxide semiconductor (CMOS) processors, implements Parallel Sysplex to provide the ultimate in redundancy and session persistence.

The FEP has often been replaced by a high-performance, channel-connected router such as the Channel Interface Processor (CIP) or the Channel Port Adapter (CPA). The network backbone comprises high-speed switches (ATM, Ethernet/Fast Ethernet/Gigabit Ethernet, or Token Ring) and routers running APPN/HPR. Shared Token Ring LANs are being replaced with Token Ring or Ethernet switching to the desktop, offering a dedicated LAN segment and bandwidth to each end user. Most desktops have PCs running advanced SNA client emulation software such as TN3270 Server. Routers provide support, via features such as Dependent Logical Unit Requester (DLUR) and downstream physical unit (DSPU) concentration, to transport the traffic from the remaining traditional SNA terminals and controllers.

Phase Two: IP Transport

Beginning in the 1980s, large organizations began building TCP/IP-based networks to support client/server applications and systems. UNIX, a dominant operating system for client/server applications, natively supports TCP/IP. As the growth of TCP/IP-based systems continued, organizations often found that they had built parallel networks, one running SNA and one running TCP/IP. This setup is expensive because of the duplication of line costs, equipment, and personnel. To eliminate the duplication, organizations had a choice—run the TCP/IP traffic over the SNA backbone, or run the SNA traffic over the TCP/IP backbone.

Running TCP/IP over an SNA backbone was not a feasible choice because of the lack of redundancy and openness of SNA. Routers, which formed the core of the TCP/IP network, began to support the encapsulation of SNA in TCP/IP for transport across the TCP/IP network using technologies such as remote source-route bridging (RSRB) and data-link switching plus (DLSw+).

This encapsulation brings many benefits. First and foremost, while it is encapsulated in TCP/IP, the SNA traffic is dynamically routed around network failures, a benefit that only recently has been added to SNA networks with APPN/HPR. The encapsulation schemes also provide more flexible configurations for SNA devices and reduced polling traffic across the backbone. Cisco offered the first such encapsulation scheme with RSRB. Since then, the industry has adopted a standard, data-link switching (DLSw), that has been very widely accepted and implemented. Routers also provide features such as serial tunnel (STUN) and Block Serial Tunneling (BSTUN) to encapsulate other types of traffic (asynchronous, bisynchronous, and some proprietary protocols) in addition to SNA.

In this second phase of integration, many organizations find that the same end users who are running advanced SNA client emulators to access mainframe and midrange systems are also accessing TCP/IP systems. This means that each PC must run two different protocol stacks—SNA and TCP/IP—for access to host systems.

Phase Three: IP Client

In the third phase of SNA-to-IP integration, organizations eliminate the dual protocol stacks at end-user PCs by implementing emulation software that supports TCP/IP. The same rich functionality that end users rely on in their emulation software remains the same, only it now runs over a TCP/IP stack. Cisco Transaction Connection (CTRC) provides TCP/IP end-users and servers with direct access to Customer Information Control System (CICS) and IBM DB2 databases. Organizations achieve protocol independence between end-users and hosts, enabling applications to communicate directly to DB2 or CICS without upgrades.

TN3270(E), TN5250, Distributed Relational Database Architecture (DRDA) and Inter-System Communications (ISC) protocol are widely implemented and widely accepted standards for achieving TCP/IP-based access to mainframes and AS/400s. The TN3270 Server technology on the router provides support for the TN3270(E) clients. CTRC on the router supports access to IBM DB2 databases from ODBC and JDBC drivers. CTRC also supports access to transaction programs managed by IBM's CICS. In addition to eliminating a second protocol from each desktop, organizations reap the following benefits by implementing low-cost, standards-based solutions such as TN3270(E), TN5250, and CTRC:

- Availability of high-performance servers. Very high-capacity and high-performance gateway servers are available that offload the protocol processing of TN3270(E) or TN5250 from the mainframe or midrange host. These servers replace the low-capacity PC gateways that are based on proprietary gateway protocols.
- Integration with corporate intranet. Because the desktop is based upon TCP/IP, all the advances taking place in corporate intranets can be brought to mainframe and midrange connections. For example, virtual private networks (VPNs) can be created for secure remote host access. Encryption and authentication can become a new level of security for host access.
- Access from a browser. A whole new market, the Web-to-host market, is emerging that allows end users to access host systems using the browser as the standard interface. This setup brings enormous benefits by reducing the software distribution and administration chores for emulation software and this sets the stage for a new, browser-style interface to older applications. Organizations can look to these mission-critical applications to extend new services to their customers, as in the case of home banking, citizen access to government records, and insurance company applications.

Phase Four: IP Centric

In the fourth and final stage of SNA-to-IP integration, the mainframe and midrange systems natively support TCP/IP. They share files with and transfer data to other, non-SNA systems. The corporate databases are securely accessed in a standard way from a variety of different end-user applications. The remaining applications that are based on traditional "green-on-black," character-based terminals are accessed transparently through standard emulation screens or through intuitive, user-friendly Web pages. These TCP/IP-based mainframe and midrange systems offer advanced redundancy and high-availability features similar to those provided to SNA-based applications today. With the full, native support of TCP/IP, the mainframe and midrange systems can be fully participating members in the corporate intranet.

Summary of Four-Phase Model

The four-phase model of SNA-to-IP integration is based on Cisco's experience helping to integrate some of the world's largest and most complex SNA networks. In reality, very few organizations go through a stepwise, linear migration from SNA centric, to IP transport, to IP client, to IP centric. For example, many large organizations have run TCP/IP stacks on their mainframes for years, alongside ACF/VTAM, whether they have implemented TCP/IP in the enterprise backbone network or not. Indeed, most large organizations will find elements from all four phases represented somewhere in their network. The model, however, is useful to describe the various issues of SNA-to-IP integration, their common solutions, and the characteristics of the network at various points in the change.

Scenarios for SNA-to-IP Integration

There are common elements or scenarios for integrating TCP/IP with SNA networks. This section describes three elements or scenarios, the corresponding phase from the Cisco four-phase integration model, and the Cisco products and software features deployed in these scenarios. This section discusses the following scenarios:

- Line Consolidation, page 5
- FEP Replacement, page 5
- Desktop Consolidation, page 6

Line Consolidation

Line consolidation involves simplifying the network by providing a single network infrastructure, based on TCP/IP. This structure accommodates SNA and other traffic and allows the elimination of multiple single-protocol lines to each location.

Phase two of SNA-to-IP integration dictates the building of a single network backbone based upon TCP/IP. This setup often allows organizations to consolidate the number of communication lines in the network which simplifies the support and maintenance.

The primary product in a line consolidation project is a multiprotocol router that encapsulates and converts the traffic from the SNA lines. RSRB and DLSw+ are the Cisco IOS technologies used for this conversion. In addition, Cisco routers also support the tunneling of both bisynchronous and certain asynchronous protocols with Cisco IOS features such as STUN and BSTUN and the Airline Product Set (ALPS).

FEP Replacement

FEP replacement involves replacing FEPs (and possibly other special-purpose mainframe channel-attached equipment) with new channel-attached routers that offer high throughput, low costs, and flexible software functionality.

Throughout all phases of the SNA-to-IP integration, high-capacity throughput to the mainframe is a key requirement. Organizations are replacing FEPs with routers with direct channel attachments.

The primary product in a FEP replacement project is a channel-attached router. This router contains the mainframe channel connection hardware supporting either a bus-and-tag or ESCON interface (or multiple interfaces). It also runs the necessary channel protocol software and, in some cases, special software designed to offload communication processing from the mainframe. For example, the Cisco CIP and CPA both support TCP Offload, TCP Assist, CTRC, and TN3270 Server features to offload mainframe cycles.

Desktop Consolidation

In a desktop consolidation, desktops running multiple protocol stacks are simplified to utilize TCP/IP for access to all resources, including mainframes and AS/400s. This consolidation can be accomplished using traditional emulators that utilize TCP/IP instead of SNA for host communication, or it can be accomplished by leveraging new browser-based access approaches.

Phases three and four of the SNA-to-IP integration require end users to access host systems using TCP/IP.

The primary products in a desktop consolidation project are desktop devices, desktop software, and new gateway servers. Other products that may be considered for deployment are additional load-balancing domain name servers, firewalls, and other security devices. Terminal emulation is, by definition, a client/server implementation. That is, PCs running terminal emulation software communicate with gateway software (located on a PC server, a router, or the host) using either a proprietary or a standard protocol that is at a higher level than the TCP/IP transport. These gateways then communicate directly with the host applications using standard SNA protocols. Most terminal emulators offer multiple choices of gateway connectivity. The only standard TCP/IP-based protocols for communication to mainframe and midrange systems are TN3270(E) and TN5250, respectively. Many organizations are implementing TN3270 and TN5250 because they are standards and they set the stage for Web-to-host solutions.

Bridging

Overview of Bridging

The Bridging section of this guide discusses the following software components for bridging and routing protocols in Cisco routers:

This overview chapter gives a high-level description of each technology. For configuration information, refer to the corresponding chapter in this publication.

Note All commands supported on the Cisco 7500 series routers are also supported on the Cisco 7000 series routers.

Transparent and Source-Route Transparent (SRT) Bridging

Cisco IOS software supports transparent bridging for Ethernet, Fiber Distributed Data Interface (FDDI), and serial media, and supports source-route transparent (SRT) bridging for Token Ring media. In addition, Cisco supports all the mandatory Management Information Base (MIB) variables specified for transparent bridging in RFC 1286.

This section contains the following topics:

Transparent Bridging Features

The Cisco transparent bridging software implementation has the following features:

- Complies with the IEEE 802.1D standard.

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

- Provides the ability to logically segment a transparently bridged network into virtual LANs.
- Provides two Spanning Tree Protocols—an older bridge protocol data unit (BPDU) format that is compatible with Digital and other LAN bridges for backward compatibility and the IEEE standard BPDU format. In addition to features standard with these Spanning Tree Protocols, the Cisco proprietary software provides for multiple domains for spanning trees. The spanning-tree parameters are configurable.
- Allows frame filtering based on Media Access Control (MAC) address, protocol type, or the vendor code. Additionally, the bridging software can be configured to selectively filter local-area transport (LAT) multicast service announcements.
- Provides deterministic load distribution while maintaining a loop-free spanning tree.
- Provides the ability to bridge over Asynchronous Transfer Mode (ATM), dial-on-demand routing (DDR), FDDI, Frame Relay, multiprotocol Link Access Procedure, Balanced (LAPB), Switched Multimegabit Data Service (SMDS), and X.25 networks.
- Provides concurrent routing and bridging, which is the ability to bridge a given protocol on some interfaces in a router and concurrently route that protocol on other interfaces in the same router.
- Provides integrated routing and bridging, which is the ability to route a given protocol between routed interfaces and bridge groups, or to route a given protocol between bridge groups.
- Provides fast-switched transparent bridging for Frame Relay encapsulated serial and High-Speed Serial Interface (HSSI) interfaces, according to the format specified in RFC 1490.
- Provides fast-switched transparent bridging for the ATM interface on the Cisco 7000, according to the format specified in RFC 1483.
- Provides for compression of LAT frames to reduce LAT traffic through the network.
- Provides both bridging and routing of VLANs.

Cisco access servers and routers can be configured to serve as both multiprotocol routers and MAC-level bridges, bridging any traffic that cannot otherwise be routed. For example, a router routing the Internet Protocol (IP) can also bridge Digital's LAT protocol or NetBIOS traffic.

Cisco routers also support remote bridging over synchronous serial lines. As with frames received on all other media types, dynamic learning and configurable filtering applies to frames received on serial lines.

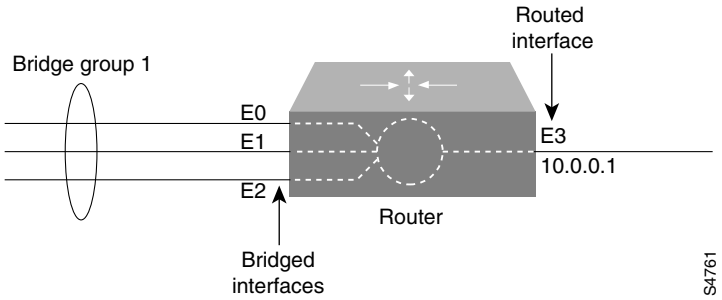
Transit bridging of Ethernet frames across FDDI media is also supported. The term *transit* refers to the fact that the source or destination of the frame cannot be on the FDDI media itself. This allows FDDI to act as a highly efficient backbone for the interconnection of many bridged networks. The configuration of FDDI transit bridging is identical to the configuration of transparent bridging on all other media types.

Integrated Routing and Bridging

Although concurrent routing and bridging makes it possible to both route and bridge a specific protocol on separate interfaces within a router, the protocol is not switched between bridged and routed interfaces. Routed traffic is confined to the routed interfaces; bridged traffic is confined to bridged interfaces. A specified protocol may be either routed or bridged on a given interface, but not both.

Integrated routing and bridging makes it possible to route a specific protocol between routed interfaces and bridge groups, or route a specific protocol between bridge groups. Local or unroutable traffic can be bridged among the bridged interfaces in the same bridge group, while routable traffic can be routed to other routed interfaces or bridge groups. Figure 2 illustrates how integrated routing and bridging in a router interconnects a bridged network with a routed network.

Figure 2 Integrated Routing and Bridging Interconnecting a Bridged Network with a Routed Network



You can configure the Cisco IOS software to route a specific protocol between routed interfaces and bridge groups or to route a specific protocol between bridge groups. Specifically, local or unroutable traffic is bridged among the bridged interfaces in the same bridge group, while routable traffic is routed to other routed interfaces or bridge groups. Using integrated routing and bridging, you can do the following:

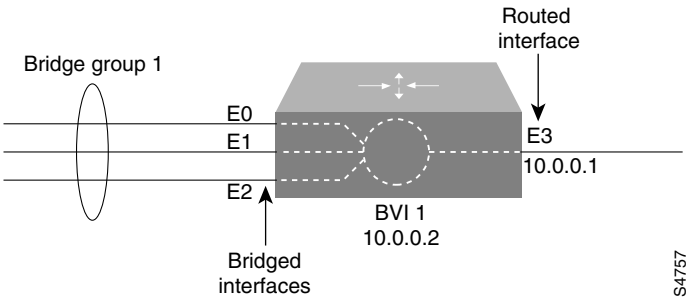
- Switch packets from a bridged interface to a routed interface
- Switch packets from a routed interface to a bridged interface
- Switch packets within the same bridge group

Bridge-Group Virtual Interface

Because bridging operates in the data link layer and routing operates in the network layer, they follow different protocol configuration models. Taking the basic IP model as an example, all bridged interfaces would belong to the same network, while each routed interface represents a distinct network.

In integrated routing and bridging, the bridge-group virtual interface is introduced to avoid confusing the protocol configuration model when a specific protocol is both bridged and routed in a bridge group. Figure 3 illustrates the bridge-group virtual interface as a user-configured virtual interface residing within a router.

Figure 3 Bridge-Group Virtual Interface in the Router



The bridge-group virtual interface is a normal routed interface that does not support bridging, but does represent its corresponding bridge group to the routed interface. It has all the network layer attributes (such as a network layer address and filters) that apply to the corresponding bridge group. The interface number assigned to this virtual interface corresponds to the bridge group that this virtual interface represents. This number is the link between the virtual interface and the bridge group.

When you enable routing for a given protocol on the bridge-group virtual interface, packets coming from a routed interface, but destined for a host in a bridged domain, are routed to the bridge-group virtual interface and are forwarded to the corresponding bridged interface. All traffic routed to the bridge-group virtual interface is forwarded to the corresponding bridge group as bridged traffic. All routable traffic received on a bridged interface is routed to other routed interfaces as if it is coming directly from the bridge-group virtual interface.

To receive routable packets arriving on a bridged interface but destined for a routed interface or to receive routed packets, the bridge-group virtual interface must also have the appropriate addresses. MAC addresses and network addresses are assigned to the bridge-group virtual interface as follows:

- The bridge-group virtual interface “borrows” the MAC address of one of the bridged interfaces in the bridge group associated with the bridge-group virtual interface.
- To route and bridge a given protocol in the same bridge group, you must configure the network layer attributes of the protocol on the bridge-group virtual interface. No protocol attributes should be configured on the bridged interfaces, and no bridging attributes can be configured on the bridge-group virtual interface.

Because there can be only one bridge-group virtual interface representing a bridge group, and the bridge group can be made up of different media types configured for several different encapsulation methods, you may need to configure the bridge-group virtual interface with the particular encapsulation methods required to switch packets correctly.

For example, the bridge-group virtual interface has default data link and network layer encapsulations that are the same as those available on Ethernet interfaces, but you can configure the bridge-group virtual interface with encapsulations that are not supported on an Ethernet interface. In some cases, the default encapsulations provide appropriate results; in other cases they do not. For example, with default encapsulation, Advanced Research Projects Agency (ARPA) packets from the bridge-group virtual interface are translated to Subnetwork Access Protocol (SNAP) when bridging IP to a Token Ring- or FDDI-bridged interface. But for Internet Packet Exchange (IPX), Novell-ether encapsulation from the bridge-group virtual interface is translated to raw-token or raw-FDDI when bridging IPX to a Token Ring- or FDDI-bridged interface. Because this behavior is usually not what you want, you must configure IPX SNAP or Service Advertisement Protocol (SAP) encapsulation on the bridge-group virtual interface.

Other Considerations

The following are additional facts regarding the support of integrated routing and bridging:

- Integrated routing and bridging is not supported on cBus platforms (AGS+ and Cisco 7000 series).
- Integrated routing and bridging is supported for transparent bridging, but not for source-route bridging (SRB).
- Integrated routing and bridging is supported on all media interfaces except X.25 and Integrated Services Digital Network (ISDN) bridged interfaces.
- Integrated routing and bridging supports three protocols: IP, IPX, and AppleTalk in both fast-switching and process-switching modes.
- Integrated routing and bridging and concurrent routing and bridging cannot operate at the same time.

SRT Bridging Features

Cisco routers support transparent bridging on Token Ring interfaces that support SRT bridging. Both transparent and SRT bridging are supported on all Token Ring interface cards that can be configured for either 4- or 16-MB transmission speeds.

As with other media, all the features that use **bridge-group** commands can be used on Token Ring interfaces. As with other interface types, the bridge group can be configured to run either the IEEE or Digital Spanning Tree Protocols. When configured for the IEEE Spanning Tree Protocol, the bridge cooperates with other SRT bridges and constructs a loop-free topology across the entire extended LAN.

You can also run the Digital Spanning Tree Protocol over Token Ring. Use it when you have other non-IEEE bridges on other media and you do not have any SRT bridges on Token Ring. In this configuration, all the Token Ring transparent bridges must be Cisco routers. This is because the Digital Spanning Tree Protocol has not been standardized on Token Ring.

As specified by the SRT bridging specification, only packets without a routing information field (RIF) (RII = 0 in the SA field) are transparently bridged. Packets with a RIF (RII = 1) are passed to the SRB module for handling. An SRT-capable Token Ring interface can have both SRB and transparent bridging enabled at the same time. However, with SRT bridging, frames that did not have a RIF when they were produced by their generating host never gain a RIF, and frames that did have a RIF when they were produced never lose that RIF.

Note Because bridges running only SRT bridging never add or remove RIFs from frames, they do not integrate SRB with transparent bridging. A host connected to a source-route bridge that expects RIFs can *never* communicate with a device across a bridge that does not understand RIFs. SRT bridging cannot tie in existing source-route bridges to a transparent bridged network. To tie in existing bridges, you must use source-route translational bridging (SR/TLB) instead. SR/TLB is described in the "Configuring Source-Route Bridging" chapter.

Bridging between Token Ring and other media requires certain packet transformations. In all cases, the MAC addresses are bit-swapped because the bit ordering on Token Ring is different from that on other media. In addition, Token Ring supports one packet format, logical link control (LLC), while Ethernet supports two formats (LLC and Ethernet).

The transformation of LLC frames between media is simple. A length field is either created (when the frame is sent to non-Token Ring) or removed (when the frame is sent to Token Ring). When an Ethernet format frame is sent to Token Ring, the frame is translated into an LLC-1 SNAP packet. The destination service access point (DSAP) value is AA, the source service access point (SSAP) value is AA, and the organizational unique identifier (OUI) value is 0000F8. Likewise, when a packet in LLC-1 format is bridged onto Ethernet media, the packet is translated into Ethernet format.

Caution Bridging between dissimilar media presents several problems that can prevent communication from occurring. These problems include bit order translation (or using MAC addresses as data), maximum transmission unit (MTU) differences, frame status differences, and multicast address usage. Some or all these problems might be present in a multimedia bridged LAN. Because of differences in the way end nodes implement Token Ring, these problems are most prevalent when bridging between Token Ring and Ethernet or between Ethernet and FDDI LANs.

Problems currently occur with the following protocols when bridged between Token Ring and other media: Novell IPX, DECnet Phase IV, AppleTalk, Banyan VINES, Xerox Network Systems (XNS), and IP. Further, problems can occur with the Novell IPX and XNS protocols when bridged between FDDI and other media. We recommend that these protocols be routed whenever possible.

Source-Route Bridging (SRB)

The Cisco IOS bridging software includes SRB capability. A source-route bridge connects multiple physical Token Rings into one logical network segment. If the network segment bridges only Token Ring media to provide connectivity, the technology is termed SRB. If the network bridges Token Ring and non-Token Ring media is introduced into the bridged network segment, the technology is termed remote source-route bridging (RSRB).

SRB enables routers to simultaneously act as a Level 3 router and a Level 2 source-route bridge. Thus, protocols such as Novell's IPX or XNS can be routed on Token Rings, while other protocols such as Systems Network Architecture (SNA) or NetBIOS are source-route bridged.

SRB technology is a combination of bridging and routing functions. A source-route bridge can make routing decisions based on the contents of the MAC frame header. Keeping the routing function at the MAC, or Level 2, layer allows the higher-layer protocols to execute their tasks more efficiently and allows the LAN to be expanded without the knowledge of the higher-layer protocols.

As designed by IBM and the IEEE 802.5 committee, source-route bridges connect extended Token Ring LANs. A source-route bridge uses the RIF in the IEEE 802.5 MAC header of a datagram (Figure 4) to determine which rings or Token Ring network segments the packet must transit.

Figure 4 IEEE 802.5 Token Ring Frame Format

The source station inserts the RIF into the MAC header immediately following the source address field in every frame, giving this style of bridging its name. The destination station reverses the routing field to reach the originating station.

The information in a RIF is derived from explorer packets generated by the source node. These explorer packets traverse the entire source-route bridge network, gathering information on the possible paths the source node might use to send packets to the destination.

Transparent spanning-tree bridging requires time to recompute a topology in the event of a failure; SRB, which maintains multiple paths, allows fast selection of alternate routes in the event of failure. Most importantly, SRB allows the end stations to determine the routes the frames take.

SRB Features

The Cisco SRB implementation has the following features:

- Provides configurable fast-switching software for SRB.
- Provides for a local source-route bridge that connects two or more Token Ring networks.
- Provides *ring groups* to configure a source-route bridge with more than two network interfaces. A ring group is a collection of Token Ring interfaces in one or more routers that are collectively treated as a *virtual ring*.
- Provides two types of explorer packets to collect RIF information—an *all-routes* explorer packet, which follows all possible paths to a destination ring, and a *spanning-tree* explorer packet, which follows a statically configured limited route (spanning tree) when looking for paths.
- Provides a dynamically determined RIF cache based on the protocol. The software also allows you to add entries manually to the RIF cache.
- Provides for filtering by MAC address, link service access point (LSAP) header, and protocol type.
- Provides for filtering of NetBIOS frames either by station name or by a packet byte offset.
- Provides for translation into transparently bridged frames to allow source-route stations to communicate with nonsource-route stations (typically on Ethernet).
- Provides support for the SRB MIB variables as described in the IETF draft "Bridge MIB" document, "Definition of Managed Objects for Bridges," by E. Decker, P. Langille, A. Rijsinghani, and K. McCloghrie, June 1991. Only the SRB component of the Bridge MIB is supported.
- Provides support for the Token Ring MIB variables as described in RFC 1231, *IEEE 802.5 Token Ring MIB*, by K. McCloghrie, R. Fox, and E. Decker, May 1991. Cisco implements the mandatory tables (Interface Table and Statistics Table), but not the optional table (Timer Table) of the Token Ring MIB. The Token Ring MIB has been implemented for the 4/16-Mb Token Ring cards that can be user adjusted for either 4- or 16-Mb transmission speeds (CSC-1R, CSC-2R, CSC-R16M, or CSC-C2CTR).
- SRB is supported over FDDI on Cisco 7200 series routers.
- Particle-based switching is supported (over FDDI and Token Ring) by default on Cisco 7200 series routers.
- Complies with RFC 1483 in Cisco IOS Release 12.0(3)T and later by offering the ability to encapsulate SRB traffic using RFC 1483 bridged LLC encapsulation. This support enables SRB over ATM functionality that is interoperable with other vendors' implementations of SRB over ATM.

Token Ring Inter-Switch Link (TRISL)

ISL is a Layer 2 protocol that enables switches and routers to transport Ethernet frames from multiple VLANs across Fast Ethernet or Gigabit Ethernet links. The Cisco TRISL protocol extends the ISL model to include the transport of Token Ring frames from multiple VLANs across these same links.

TRISL support on Cisco routers provides inter-VLAN routing and bridging across a 100 Mb Fast Ethernet link. ISL and TRISL together provide routing and bridging between Token Ring and Ethernet LANs, ELANS, and VLANs.

TRISL is supported on the following platforms with any one of the following port adapters:

- Cisco 7500 or Cisco 7200 series routers
  - Two-port Fast Ethernet/ISL 100BaseTX
  - Two-port Fast Ethernet/ISL 100BaseFX
  - One-port Fast Ethernet 100BaseTX
  - One-port Fast Ethernet 100BaseFX
- Cisco 4500 or 4700 series routers
  - NM-1FE
- Cisco 3600 or 2600 series routers
  - NM-1FE1CT1
  - NM-1FE2CT1
  - NM-1FE1CE1

Note The two-port Fast Ethernet/ISL port adapters support frame sizes up to 17800 bytes and the one-port Fast Ethernet port adapters support a frame size of up to 1500 bytes.

TRISL provides the following new capabilities and features, which are described in the "TRISL Configuration Task List" section on page 148 and in the "TRISL Configuration Examples" section on page 154:

- IP routing for source-routed and nonsource-routed frames between TRISL VLANs and any LAN, ELAN, or VLAN.
- IPX routing for source-routed and nonsource-routed frames between TRISL VLANs and any LANs, ELANs, or VLANs.
- SRB between TRISL VLANs and SRB-capable LANs, ELANs, or VLANs.
- SRT between TRISL VLANs and SRT-capable LANs, ELANs, or VLANs.
- SR/TLB between TRISL VLANs and Ethernet LANs, ELANs, or VLANs.
- Duplicate Ring Protocol (DRiP), which prevents external loops that could result if the router's virtual ring number were duplicated elsewhere in the network.

Note VLAN Trunk Protocol (VTP) is not supported for TRISL on the routers in this release.

Token Ring Route Switch Module (TRRSM)

The Token Ring VLAN support on the Route Switch Module (RSM) adds the capability to do multi-protocol routing and bridging for Token Ring VLANs on the RSM. The RSM is a router module running Cisco IOS software that plugs into a Token Ring switch backplane.

This section contains a brief overview of Token Ring switching, which is described in the following topics:

Switching Overview

The term switching was originally used to describe packet-switch technologies such as Link Access Procedure, Balanced (LAPB), Frame Relay, Switched Multimegabit Data Service (SMDS), and X.25. Today, LAN switching refers to a technology that is similar to a bridge in many ways.

Like bridges, switches connect LAN segments and use information contained in the frame to determine the segment to which a datagram needs to be sent. Switches, however, operate at much higher speeds than bridges, and can support new functionality, such as virtual LANs (VLANs). See the "VLAN" section on page 10 and the "Token Ring VLANs" section on page 10.

Token Ring switches first appeared in 1994. The first-generation Token Ring switches can be divided into two basic categories:

- Processor-based switches—These switches use reduced instruction set computer (RISC) processors to switch Token Ring frames. Although they typically have a lot of function, they are slow and relatively expensive. These switches have been deployed mainly as backbone switches because of their high cost.
- Application-specific integrated circuit (ASIC)-based switches with limited functionality—These switches are fast and relatively inexpensive, but have very limited function. Typically, they offer little to no filtering, limited management information, limited support for bridging modes, and limited VLANs. Today, although these switches are less expensive than processor-based switches, they are still too expensive and limited for widespread use of dedicated Token Ring to the desktop.

In 1997, a second generation of Token Ring switches was introduced. The Cisco second-generation Token Ring switches use ASIC-based switching, but they provide increased functionality resulting in a higher speed and lower cost. They also provide a wider variety of function than their predecessors, including support for multiple bridging modes, Dedicated Token Ring (DTR) on all ports, high-port density, high-speed links, filtering, Remote Monitoring (RMON) management, broadcast control, and flexible VLANs.

The family of second-generation Token Ring switches can be used for backbone switching, workgroup microsegmentation, and dedicated Token Ring to the desktop. Token Ring switches currently being offered include:

- The Catalyst 3900, which is a stackable workgroup switch that provides support for all switching modes, filtering, RMON, DTR, and SNMP management, and support for ATM and Inter-Switch Link (ISL).
- The Catalyst 3920, which is a also a stackable workgroup switch that provides support for all switching modes, filtering, RMON, DTR, and SNMP management.
- The Catalyst 5000, which is a modular switch that supports Ethernet, Fast Ethernet, FDDI, ATM, and now Token Ring.

The Catalyst Token Ring switches support the following bridging modes: SRB, SRT, and source-route switching.

Usability of Switching

The traditional method of connecting multiple Token Ring segments is to use a SRB. For example, bridges are often used to link workgroup rings to the backbone ring. However, the introduction of the bridge can significantly reduce performance at the user's workstation. Further problems may be introduced by aggregate traffic loading on the backbone ring.

To maintain performance and avoid overloading the backbone ring, you can locate servers on the same ring as the workgroup that needs to access the server. However, dispersing the servers throughout the network makes them more difficult to back up, administer, and secure than if they are located on the backbone ring. Dispersing the servers also limits the number of servers that particular stations can access.

Collapsed backbone routers may offer greater throughput than bridges, and can interconnect a larger number of rings without becoming overloaded. Routers provide both bridging and routing functions between rings and have sophisticated broadcast control mechanisms. These mechanisms become increasingly important as the number of devices on the network increases.

The main drawback of using routers as the campus backbone is the relatively high price per port and the fact that the throughput typically does not increase as ports are added. A Token Ring switch is designed to provide wire speed throughput regardless of the number of ports in the switch. In addition, the Catalyst 3900 Token Ring switch can be configured to provide very low latency between Token Ring ports by using cut-through switching.

As a local collapsed backbone device, a Token Ring switch offers a lower per-port cost and can incur lower interstation latency than a router. In addition, the switch can be used to directly attach large numbers of clients or servers, thereby replacing concentrators. Typically, a Token Ring switch is used in conjunction with a router, providing a high-capacity interconnection between Token Ring segments while retaining the broadcast control and wide-area connectivity provided by the router.

VLAN

A VLAN is a logical group of LAN segments, independent of physical location, with a common set of requirements. For example, several end stations might be grouped as a department, such as engineering or accounting. If the end stations are located close to one another, they can be grouped into a LAN segment. If any of the end stations are on a different LAN segment, such as different buildings or locations, they can be grouped into a VLAN that has the same attributes as a LAN even though the end stations are not all on the same physical segment. The information identifying a packet as part of a specific VLAN is preserved across a Catalyst switch connection to a router or another switch if they are connected via trunk ports, such as ISL or ATM.

Token Ring VLANs

Because a VLAN is essentially a broadcast domain, a Token Ring VLAN is slightly more complex than an Ethernet VLAN. In transparent bridging, there is only one type of broadcast frame and, therefore, only one level of broadcast domain and one level of VLAN. In source routing, however, there are two types of broadcast frames:

- Those that are confined to a single ring
- Those that traverse the bridged domain

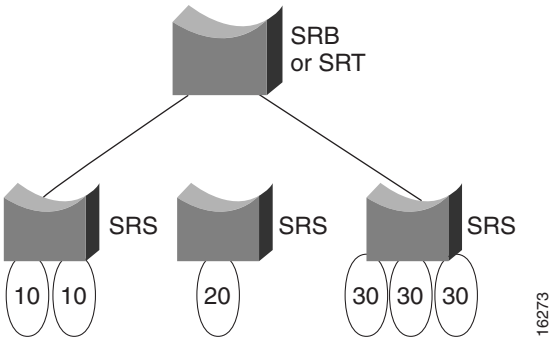
Therefore, there are two levels of VLANs in a Token Ring switched network. These two categories of broadcast frames result in a broadcast domain that is hierarchical in nature, just as a local ring domain can exist only within a domain of all the interconnected rings.

The first level is the Token Ring Concentrator Relay Function (TrCRF). In a Token Ring VLAN, logical ring domains are formed by defining groups of ports that have the same ring number. The IEEE calls such a port group a Concentrator Relay Function (CRF). On Catalyst switches, such a grouping of Token Ring ports is called a Token Ring CRF (TrCRF). At this level, the VLAN is a logical ring and, as such, is assigned a ring number. On a Token Ring switch, the logical ring (TrCRF) contains one or more physical ports. source-route switching is used to forward frames within a TrCRF based on MAC address or Route Descriptor. On an RSM, a logical ring (TrCRF) can be defined that does not contain any physical ports, but rather is used only in processing source-routed traffic to terminate the RIF.

The second level of VLAN is the Token Ring Bridge Relay Function (TrBRF). This is the parent VLAN to which TrCRF VLANs are assigned. The domain of interconnected rings is formed using an internal multiport bridge function that the IEEE calls a Bridge Relay Function (BRF). On Catalyst switches, such a grouping of logical rings is called a Token Ring BRF (TrBRF). At this level, the VLAN is a logical bridge and, as such, is assigned a bridge number. The TrBRF is responsible for forwarding frames between groups of ports with the same ring number (TrCRFs) via either SRB or SRT.

Figure 5 depicts the relationship between TrCRF and TrBRF VLANs.

Figure 5 Token Ring VLAN Support on the RSM



Token Ring VLAN Support on the RSM

The Token Ring VLAN support on the RSM adds the capability to do multiprotocol routing and bridging for Token Ring VLANs on the RSM. The RSM can be used alone to do inter-VLAN routing, or it can be paired with a Catalyst VIP2 to provide external network connections with the same port adapters used on Cisco 7500 series routers. The RSM/VIP2 combination provides routing between VLANs and Catalyst VIP2 port adapters. A complete description of the RSM can be found in the *Catalyst 5000 Series Route Switch Module Installation and Configuration Note* and the *Route Switch Module Catalyst VIP2-15 and VIP2-40 Installation and Configuration Note*.

The Token Ring VLAN support on the RSM adds the following functionality to the Catalyst 5000 switch:

- IP routing for source-routed and non-source-routed frames between Token Ring (TrBRF) or Ethernet VLANs and VIP2 interfaces
- IPX routing for source-routed and non-source-routed frames between Token Ring (TrBRF) or Ethernet VLANs and VIP2 interfaces
- SRB between Token Ring (TrBRF) VLANs and VIP2 interfaces

- SR/TLB between Token Ring (TrBRF) VLANs and Ethernet VLANs and VIP2 interfaces
- SRT between Token Ring (TrBRF) VLANs and SRT-capable VLANs and VIP2 interfaces

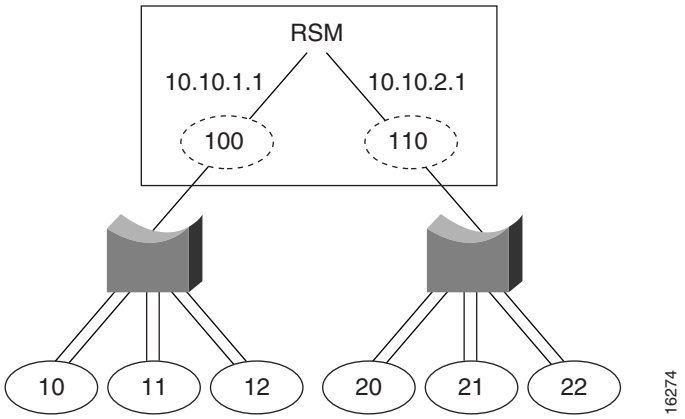
Both APPN and DLSw+ are supported for Token Ring VLANs on the RSM. However, RSRB is not supported on the RSM.

For information on how Token Ring VLANs are implemented on switches, refer to the *Catalyst Token Ring Switching Implementation Guide,* the *Catalyst 5000 Series Token Ring Configuration Notes,* the *Catalyst 3900 Token Ring Switching User Guide,* and the *Catalyst 3920 Token Ring Switching User Guide*.

The RSM is a router module running Cisco IOS router software that directly interfaces (plugs into) the Catalyst switch backplane. From the Token Ring VLAN perspective, the interface to the RSM is at the Token Ring bridged network (TrBRF) level. With the RSM, it is possible to route or bridge between separate Token Ring and Ethernet domains.

When routing or bridging between TrBRF VLANs that are defined as SRB domains, it is necessary to create a logical ring on the RSM for proper RIF processing. This logical ring is defined as a TrCRF VLAN that does not contain any external Token Ring switch ports. Figure 6 illustrates the logical view of IP routing between two source-route bridged VLANs on the RSM. In this view, the RSM appears to have an interface to both ring 100 and ring 110.

Figure 6 Logical View of VLAN Support on the RSM



The Token Ring RSM feature is supported on the RSM in the Catalyst 5000 platform. Support for the Token Ring RSM feature was first introduced in the Cisco IOS Release 11.3(5)T. The Token Ring RSM feature is supported on all Cisco IOS Release 12.0 T software release images. A list of the supported Cisco IOS releases and software images are located in the *Release Notes for Catalyst 5000 Series RSM/VIP2 Cisco IOS 12.0 T Software Releases* publication. For a complete functional description of the RSM, refer to the *Catalyst 5000 Series RSM Installation and Configuration Note*.

Configuring Transparent Bridging

Configuring Transparent Bridging

The Cisco IOS software bridging functionality combines the advantages of a spanning-tree bridge and a full multiprotocol router. This combination provides the speed and protocol transparency of an adaptive spanning-tree bridge, along with the functionality, reliability, and security of a router.

This chapter describes how to configure transparent bridging and source-route transparent (SRT) bridging. This chapter also describes the concepts of virtual networking, transparent bridging of virtual LANs (VLANs), and routing between VLANs. For a complete description of the transparent bridging commands mentioned in this chapter, refer to the "Transparent Bridging Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

Technology Overview

The following sections provide an overview of transparent bridging in the Cisco IOS software:

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

Transparent and SRT Bridging

Cisco IOS software supports transparent bridging for Ethernet, Fiber Distributed Data Interface (FDDI), and serial media, and supports source-route transparent (SRT) bridging for Token Ring media. In addition, Cisco supports all the mandatory Management Information Base (MIB) variables specified for transparent bridging in RFC 1286.

Transparent Bridging Features

Cisco's transparent bridging software implementation has the following features:

- Complies with the IEEE 802.1D standard.
- Provides the ability to logically segment a transparently bridged network into virtual LANs.
- Provides two Spanning Tree Protocols—an older bridge protocol data unit (BPDU) format that is compatible with Digital Equipment Corporation (DEC) and other LAN bridges for backward compatibility and the IEEE standard BPDU format. In addition to features standard with these Spanning Tree Protocols, Cisco's proprietary software provides for multiple domains for spanning trees. The spanning-tree parameters are configurable.
- Allows frame filtering based on Media Access Control (MAC) address, protocol type, or the vendor code. Additionally, the bridging software can be configured to selectively filter local-area transport (LAT) multicast service announcements.
- Provides deterministic load distribution while maintaining a loop-free spanning tree.
- Provides the ability to bridge over Asynchronous Transfer Mode (ATM), dial-on-demand routing (DDR), FDDI, Frame Relay, multiprotocol Link Access Procedure, Balanced (LAPB), Switched Multimegabit Data Service (SMDS), and X.25 networks.
- Provides concurrent routing and bridging, which is the ability to bridge a given protocol on some interfaces in a router and concurrently route that protocol on other interfaces in the same router.
- Provides integrated routing and bridging, which is the ability to route a given protocol between routed interfaces and bridge groups, or to route a given protocol between bridge groups.
- Provides fast-switched transparent bridging for Frame Relay encapsulated serial and High-Speed Serial Interface (HSSI) interfaces, according to the format specified in RFC 1490.
- Provides fast-switched transparent bridging for the ATM interface on Cisco 7000 series routers, Cisco 4500, and Cisco 4000 series routers, according to the format specified in RFC 1483.
- Provides for compression of LAT frames to reduce LAT traffic through the network.
- Provides both bridging and routing of VLANs.

Cisco access servers and routers can be configured to serve as both multiprotocol routers and MAC-level bridges, bridging any traffic that cannot otherwise be routed. For example, a router routing the IP can also bridge DEC's LAT protocol or NetBIOS traffic.

Cisco routers also support remote bridging over synchronous serial lines. As with frames received on all other media types, dynamic learning and configurable filtering applies to frames received on serial lines.

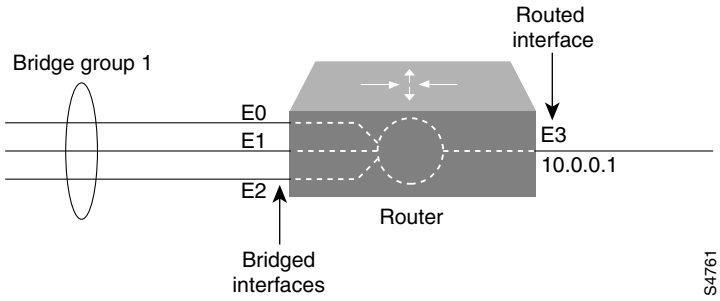
Transit bridging of Ethernet frames across FDDI media is also supported. The term *transit* refers to the fact that the source or destination of the frame cannot be on the FDDI media itself. This allows FDDI to act as a highly efficient backbone for the interconnection of many bridged networks. The configuration of FDDI transit bridging is identical to the configuration of transparent bridging on all other media types.

Integrated Routing and Bridging

While concurrent routing and bridging makes it possible to both route and bridge a specific protocol on separate interfaces within a router, the protocol is not switched between bridged and routed interfaces. Routed traffic is confined to the routed interfaces; bridged traffic is confined to bridged interfaces. A specified protocol may be either routed or bridged on a given interface, but not both.

Integrated routing and bridging makes it possible to route a specific protocol between routed interfaces and bridge groups, or route a specific protocol between bridge groups. Local or unroutable traffic can be bridged among the bridged interfaces in the same bridge group, while routable traffic can be routed to other routed interfaces or bridge groups. Figure 7 illustrates how integrated routing and bridging in a router interconnects a bridged network with a routed network.

Figure 7 Integrated Routing and Bridging Interconnecting a Bridged Network with a Routed Network



You can configure the Cisco IOS software to route a specific protocol between routed interfaces and bridge groups or to route a specific protocol between bridge groups. Specifically, local or unroutable traffic is bridged among the bridged interfaces in the same bridge group, while routable traffic is routed to other routed interfaces or bridge groups. Using integrated routing and bridging, you can do the following:

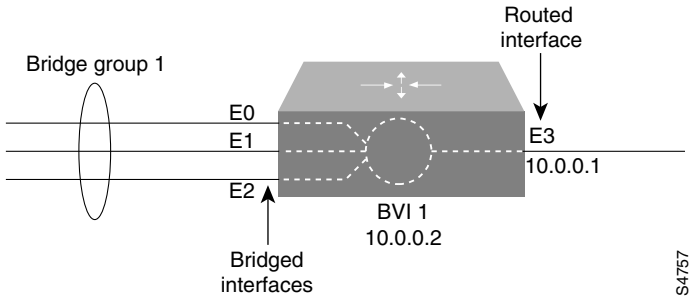
- Switch packets from a bridged interface to a routed interface
- Switch packets from a routed interface to a bridged interface
- Switch packets within the same bridge group

Bridge-Group Virtual Interface

Because bridging operates in the data link layer and routing operates in the network layer, they follow different protocol configuration models. Taking the basic IP model as an example, all bridged interfaces would belong to the same network, while each routed interface represents a distinct network.

In integrated routing and bridging, the bridge-group virtual interface is introduced to avoid confusing the protocol configuration model when a specific protocol is both bridged and routed in a bridge group. Figure 8 illustrates the bridge-group virtual interface as a user-configured virtual interface residing within a router.

Figure 8 Bridge-Group Virtual Interface in the Router



The bridge-group virtual interface is a normal routed interface that does not support bridging, but does represent its corresponding bridge group to the routed interface. It has all the network layer attributes (such as a network layer address and filters) that apply to the corresponding bridge group. The interface number assigned to this virtual interface corresponds to the bridge group that this virtual interface represents. This number is the link between the virtual interface and the bridge group.

When you enable routing for a given protocol on the bridge-group virtual interface, packets coming from a routed interface, but destined for a host in a bridged domain, are routed to the bridge-group virtual interface and are forwarded to the corresponding bridged interface. All traffic routed to the bridge-group virtual interface is forwarded to the corresponding bridge group as bridged traffic. All routable traffic received on a bridged interface is routed to other routed interfaces as if it is coming directly from the bridge-group virtual interface.

To receive routable packets arriving on a bridged interface but destined for a routed interface or to receive routed packets, the bridge-group virtual interface must also have the appropriate addresses. MAC addresses and network addresses are assigned to the bridge-group virtual interface as follows:

- The bridge-group virtual interface “borrows” the MAC address of one of the bridged interfaces in the bridge group associated with the bridge-group virtual interface.
- To route and bridge a given protocol in the same bridge group, you must configure the network layer attributes of the protocol on the bridge-group virtual interface. No protocol attributes should be configured on the bridged interfaces, and no bridging attributes can be configured on the bridge-group virtual interface.

Note When a bridged domain contains learning devices (such as switches or bridges) that can learn the MAC address of a bridge-group virtual interface, the virtual interface must be configured with its own MAC address—separate from the MAC addresses of the bridged interfaces in the bridge group that are associated with the virtual interface. The MAC address is configured by using the **mac-address** virtual interface command.

Because there can be only one bridge-group virtual interface representing a bridge group, and the bridge group can be made up of different media types configured for several different encapsulation methods, you may need to configure the bridge-group virtual interface with the particular encapsulation methods required to switch packets correctly.

For example, the bridge-group virtual interface has default data link and network layer encapsulations that are the same as those available on Ethernet interfaces, but you can configure the bridge-group virtual interface with encapsulations that are not supported on an Ethernet interface. In some cases, the default encapsulations provide appropriate results; in other cases they do not. For example, with default encapsulation, Advanced Research Projects Agency (ARPA) packets from the bridge-group virtual interface are translated to Subnetwork Access Protocol (SNAP) when bridging IP to a Token Ring- or FDDI-bridged interface. But for Internet Packet Exchange (IPX), Novell-ether encapsulation from the

bridge-group virtual interface is translated to raw-token or raw-FDDI when bridging IPX to a Token Ring- or FDDI-bridged interface. Because this behavior is usually not what you want, you must configure IPX SNAP or Service Advertisement Protocol (SAP) encapsulation on the bridge-group virtual interface.

Other Considerations

The following are additional facts regarding the support of integrated routing and bridging:

- Integrated routing and bridging is not supported on cBus platforms (AGS+ and Cisco 7000 series routers).
- Integrated routing and bridging is supported for transparent bridging, but not for source-route bridging (SRB).
- Integrated routing and bridging is supported on all media interfaces except X.25 and Integrated Services Digital Network (ISDN) bridged interfaces.
- Integrated routing and bridging supports three protocols: IP, IPX, and AppleTalk in both fast-switching and process-switching modes.
- Integrated routing and bridging and concurrent routing and bridging cannot operate at the same time.
- With integrated routing and bridging configured, associate Layer-3 attributes only on the bridge-group virtual interface and not on the bridging interfaces. Having IP addresses both on the bridge-group virtual interface and on the bridging interfaces is known to produce inconsistent behavior.
- The IEEE 802.1Q standard enables integrated routing and bridging to support connectivity for multiple VLANs using a Bridge-Group Virtual Interface (BVI) to associate a bridge group.

SRT Bridging Features

Cisco routers support transparent bridging on Token Ring interfaces that support SRT bridging. Both transparent and SRT bridging are supported on all Token Ring interface cards that can be configured for either 4- or 16-MB transmission speeds.

As with other media, all the features that use **bridge-group** commands can be used on Token Ring interfaces. As with other interface types, the bridge group can be configured to run either the IEEE or DEC Spanning Tree Protocols. When configured for the IEEE Spanning Tree Protocol, the bridge cooperates with other SRT bridges and constructs a loop-free topology across the entire extended LAN.

You can also run the DEC Spanning Tree Protocol over Token Ring. Use it when you have other non-IEEE bridges on other media and you do not have any SRT bridges on Token Ring. In this configuration, all the Token Ring transparent bridges must be Cisco routers. This is because the DEC Spanning Tree Protocol has not been standardized on Token Ring.

As specified by the SRT bridging specification, only packets without a routing information field (RIF) (RII = 0 in the SA field) are transparently bridged. Packets with a RIF (RII = 1) are passed to the SRB module for handling. An SRT-capable Token Ring interface can have both SRB and transparent bridging enabled at the same time. However, with SRT bridging, frames that did not have a RIF when they were produced by their generating host never gain a RIF, and frames that did have a RIF when they were produced never lose that RIF.

Note Because bridges running only SRT bridging never add or remove RIFs from frames, they do not integrate SRB with transparent bridging. A host connected to a source-route bridge that expects RIFs can *never* communicate with a device across a bridge that does not understand RIFs. SRT bridging cannot tie in existing source-route bridges to a transparent bridged network. To tie in existing bridges, you must use source-route translational bridging (SR/TLB) instead. SR/TLB is described in the chapter "Configuring Source-Route Bridging."

Bridging between Token Ring and other media requires certain packet transformations. In all cases, the MAC addresses are bit-swapped because the bit ordering on Token Ring is different from that on other media. In addition, Token Ring supports one packet format, logical link control (LLC), while Ethernet supports two formats (LLC and Ethernet).

The transformation of LLC frames between media is simple. A length field is either created (when the frame is sent to non-Token Ring) or removed (when the frame is sent to Token Ring). When an Ethernet format frame is sent to Token Ring, the frame is translated into an LLC-1 SNAP packet. The destination service access point (DSAP) value is AA, the source service access point (SSAP) value is AA, and the organizational unique identifier (OUI) value is 0000F8. Likewise, when a packet in LLC-1 format is bridged onto Ethernet media, the packet is translated into Ethernet format.

Caution Bridging between dissimilar media presents several problems that can prevent communication from occurring. These problems include bit order translation (or using MAC addresses as data), maximum transmission unit (MTU) differences, frame status differences, and multicast address usage. Some or all these problems might be present in a multimedia bridged LAN. Because of differences in the way end nodes implement Token Ring, these problems are most prevalent when bridging between Token Ring and Ethernet or between Ethernet and FDDI LANs.

Problems currently occur with the following protocols when bridged between Token Ring and other media: Novell IPX, DECnet Phase IV, AppleTalk, Banyan VINES, Xerox Network Systems (XNS), and IP. Further, problems can occur with the Novell IPX and XNS protocols when bridged between FDDI and other media. We recommend that these protocols be routed whenever possible.

Transparent and SRT Bridging Configuration Task List

To configure transparent bridging or SRT bridging on your router, complete one or more of the tasks in the following sections:

- Configuring Transparent and IRB Bridging on a PA-12E/2FE Ethernet Switch, page 33

See the "Transparent and SRT Bridging Configuration Examples" section on page 40 for examples.

Configuring Transparent Bridging and SRT Bridging

To configure transparent and SRT bridging, you must perform the following tasks:

Assigning a Bridge Group Number and Defining the Spanning Tree Protocol

The first step in setting up your transparent bridging network is to define a Spanning Tree Protocol and assign a bridge group number. You can choose either the IEEE 802.1D Spanning Tree Protocol, the earlier DEC protocol upon which this IEEE standard is based or VLAN bridge Spanning Tree Protocol. Cisco expanded the original 802.1 D Spanning Tree Protocol by providing VLAN bridge Spanning Tree Protocol support and increased port identification capability. Furthermore, the enhancement provides:

- More than one byte on a port number to distinguish interfaces
- An improved way to form the port ID

Port Number size of the Port ID support is applied only to IEEE and VLAN-bridge Spanning Tree Protocols. The DEC protocol only has 8 bits on the Port ID, so the extension of the Port ID cannot be applied.

The expansion of the Port Number field into the port priority portion of the Port ID changes the useful values the port priority can be assigned.

The way to calculate the Port Path Cost is only supported in IEEE and VLAN-bridge Spanning Tree Protocol environment.

To assign a bridge group number and define a Spanning Tree Protocol, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `protocol {ieee | dec | vlan-bridge}`	Assigns a bridge group number and defines a Spanning Tree Protocol as IEEE 802.1D standard, DEC or VLAN bridge.

The IEEE 802.1D Spanning Tree Protocol is the preferred way of running the bridge. Use the DEC Spanning Tree Protocol only for backward compatibility. The VLAN-bridge Spanning Tree Protocol supports the following media: Ethernet, Fast Ethernet, FDDI, ATM and serial (HDLC, PPP, Frame Relay IETF, SMDS, X.25).

Assigning Each Network Interface to a Bridge Group

A bridge group is an internal organization of network interfaces on a router. Bridge groups cannot be used outside the router on which it is defined to identify traffic switched within the bridge group. Bridge groups within the same router function as distinct bridges; that is, bridged traffic and bridge protocol data units (BPDUs cannot be exchanged between different bridge groups on a router. Furthermore, bridge groups cannot be used to multiplex or de-multiplex different streams of bridged traffic on a LAN. An interface can be a member of only one bridge group. Use a bridge group for each separately bridged (topologically distinct) network connected to the router. Typically, only one such network exists in a configuration.

The purpose of placing network interfaces into a bridge group is twofold:

- To bridge all nonrouted traffic among the network interfaces making up the bridge group. If the packet's destination address is known in the bridge table, it is forwarded on a single interface in the bridge group. If the packet's destination is unknown in the bridge table, it is flooded on all forwarding interfaces in the bridge group. The bridge places source addresses in the bridge table as it learns them during the process of bridging.
- To participate in the spanning-tree algorithm by receiving, and in some cases sending, BPDUs on the LANs to which they are attached. A separate spanning process runs for each configured bridge group. Each bridge group participates in a separate spanning tree. A bridge group establishes a spanning tree based on the BPDUs it receives on only its member interfaces.

For SRT bridging, if the Token Ring and serial interfaces are in the same bridge group, changing the serial encapsulation method causes the state of the corresponding Token Ring interface to be reinitialized. Its state will change from "up" to "initializing" to "up" again within a few seconds.

After you assign a bridge group number and define a Spanning Tree Protocol, assign each network interface to a bridge group by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group*	Assigns a network interface to a bridge group.

Choosing the OUI for Ethernet Type II Frames

For SRT bridging networks, you must choose the organizational unique identifier (OUI) code that will be used in the encapsulation of Ethernet Type II frames across Token Ring backbone networks. To choose the OUI, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# ethernet-transit-oui [90-compatible | standard | cisco]`	Selects the Ethernet Type II OUI encapsulation code.

Transparently Bridged VLANs for ISL

Traditionally, a bridge group is an independently bridged subnetwork. In this definition, bridge groups cannot exchange traffic with other bridge groups, nor can they multiplex or de-multiplex different streams of bridged traffic. The transparently bridged VLAN feature in Cisco IOS software permits a bridge group to extend outside the router to identify traffic switched within the bridge group.

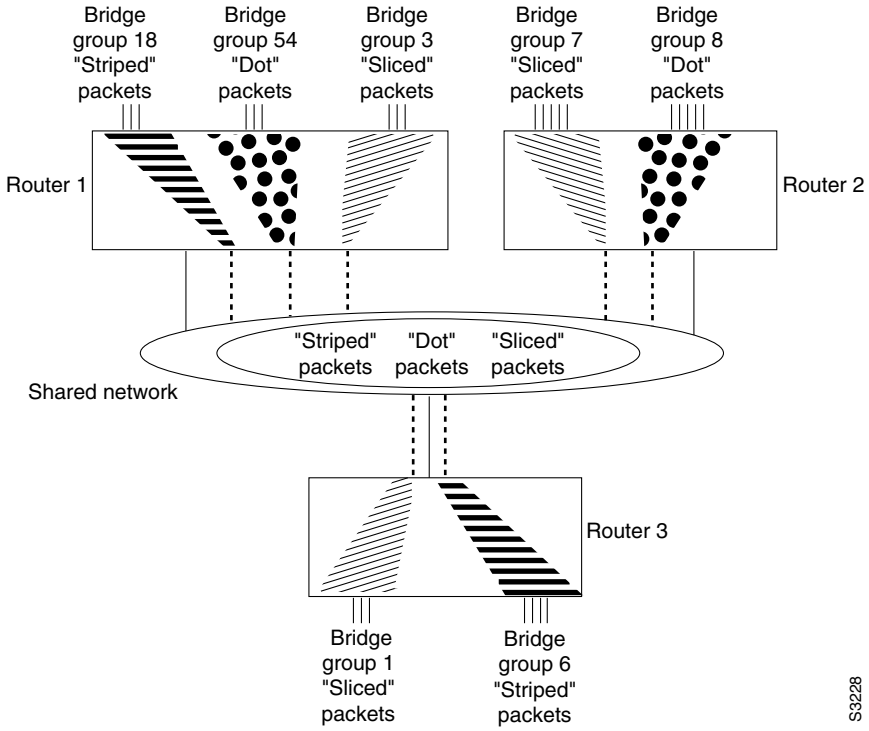
While bridge groups remain internal organizations of network interfaces functioning as distinct bridges within a router, transparent bridging on subinterfaces permits bridge groups to be used to multiplex different streams of bridged traffic on a LAN or HDLC serial interface. In this way, bridged traffic may be switched out of one bridge group on one router, multiplexed across a subinterface, and demultiplexed into a second bridge group on a second router. Together, the first bridge group and the second bridge group form a transparently bridged VLAN. This approach can be extended to impose logical topologies upon transparently bridged networks.

The primary application of transparently bridged VLANs constructed in this way is to separate traffic between bridge groups of local network interfaces, to multiplex bridged traffic from several bridge groups on a shared interface (LAN or HDLC serial), and to form VLANs composed of collections of bridge groups on several routers. These VLANs improve performance because they reduce the propagation of locally bridged traffic, and they improve security benefits because they completely separate traffic.

In Figure 9, different bridge groups on different routers are configured into three VLANs that span the bridged network. Each bridge group consists of conventionally bridged local interfaces and a subinterface on the backbone FDDI LAN. Bridged traffic on the subinterface is encapsulated and "colored" with a VLAN identifier known as a *security association identifier* common to all bridge groups participating in the VLAN. In addition, bridges only accept packets bearing security association identifiers for which they have a configured subinterface. Thus, a bridge group is configured to participate in a VLAN if it contains a subinterface configured with the VLAN's characteristic security association identifier. See the "Complex Integrated Routing and Bridging Example" section on page 43 for an example configuration of the topology shown in Figure 9.

Note The 802.10 encapsulation used to "color" transparently bridged packets on subinterfaces might increase the size of a packet so that it exceeds the MTU size of the LAN from which the packet originated. To avoid MTU violations on the shared network, the originating LANs must either have a smaller native MTU than the shared network (as is the case from Ethernet to FDDI), or the MTU on all packet sources on the originating LAN must be configured to be at least 16 bytes less than the MTU of the shared network.

Figure 9 Transparently Bridged VLANs on an FDDI Backbone



To configure a VLAN on a transparently bridged network, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type slot/port.subinterface-number`	Specifies a subinterface.
Step 2	`Router(config-if)# encapsulation sde said`	Specifies the IEEE 802.10 Security data exchange security association identifier (in other words, specifies the "color").
Step 3	`Router(config-if)# bridge-group bridge-group`	Associates the subinterface with an existing bridge group.

Transparently bridged VLANs are supported in conjunction with only the IEEE Spanning Tree Protocol. When you logically segment a transparently bridged network into VLANs, each VLAN computes its own spanning-tree topology. Configuring each VLAN to compute its own spanning-tree topology provides much greater stability than running a single spanning tree throughout. Traffic bridged within one VLAN is unaffected by physical topology changes occurring within another VLAN.

Note The current implementation of SDE encapsulation is not recommended for serial or Ethernet media.

Routing between ISL VLANs

Our VLAN Routing implementation is designed to operate across all router platforms. However, the Inter-Switch Link (ISL) VLAN trunking protocol currently is defined on 100 BaseTX/FX Fast Ethernet interfaces only and therefore is appropriate to the Cisco 7000 and higher-end platforms only. The IEEE 802.10 protocol can run over any LAN or HDLC serial interface. VLAN traffic is fast switched. The actual format of these VLAN encapsulations are detailed in the *IEEE Standard 802.10-1992 Secure Data Exchange* and in the *Inter-Switch Link (ISL) Protocol Specification*.

Our VLAN Routing implementation treats the ISL and 802.10 protocols as encapsulation types. On a physical router interface that receives and sends VLAN packets, you can select an arbitrary subinterface and map it to the particular VLAN "color" embedded within the VLAN header. This mapping allows you to selectively control how LAN traffic is routed or switched outside of its own VLAN domain. In the VLAN routing paradigm, a switched VLAN corresponds to a single routed subnet, and the network address is assigned to the subinterface.

To route a received VLAN packet the Cisco IOS software VLAN switching code first extracts the VLAN ID from the packet header (this is a 10-bit field in the case of ISL and a 4-byte entity known as the security association identifier in the case of IEEE 802.10), then demultiplexes the VLAN ID value into a subinterface of the receiving port. If the VLAN color does not resolve to a subinterface, the Cisco IOS software can transparently bridge the foreign packet natively (without modifying the VLAN header) on the condition that the Cisco IOS software is configured to bridge on the subinterface itself. For VLAN packets that bear an ID corresponding to a configured subinterface, received packets are then classified by protocol type before running the appropriate protocol specific fast switching engine. If the subinterface is assigned to a bridge group then non-routed packets are de-encapsulated before they are bridged. This is termed "fall-back bridging" and is most appropriate for nonroutable traffic types.

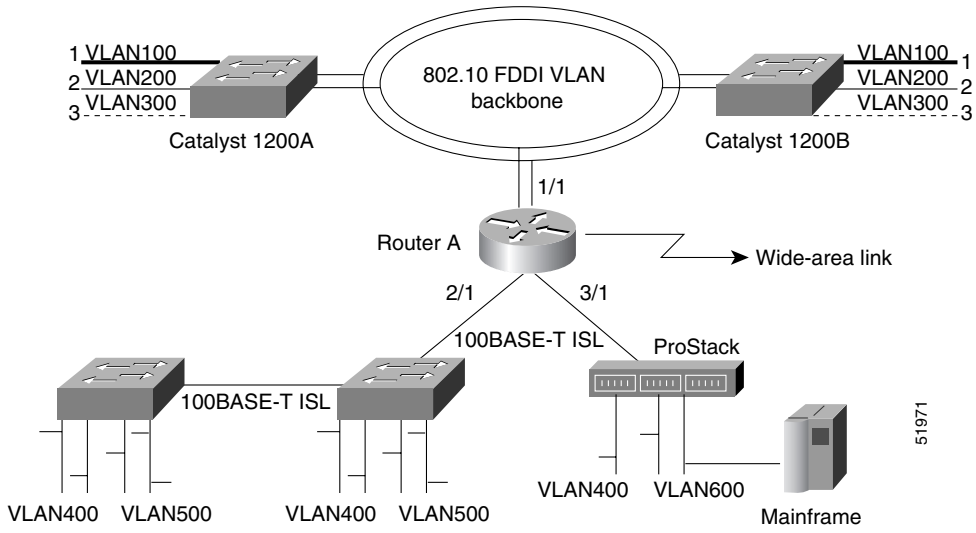
In Figure 10, Router A provides inter-VLAN connectivity between multiple Cisco switching platforms where there are three distinct virtual topologies present. For example, for VLAN 300 across the two Catalyst 1200A segments, traffic originating on LAN interface 1 is "tagged" with a VLAN ID of 300 as it is switched onto the FDDI ring. This ID allows the remote Catalyst 1200A to make an intelligent forwarding decision and only switch the traffic to local interfaces configured as belonging to the same

VLAN broadcast domain. Router A provides an inter-VLAN mechanism that lets Router A function as a gateway for stations on a given LAN segment by sending VLAN encapsulated traffic to and from other switched VLAN domains or simply sending traffic in native (non-VLAN) format.

Figure 10 illustrates the following scenarios:

- Clients on VLAN 300 want to establish sessions with a server attached to a port in a different VLAN (600). In this scenario, packets originating on LAN interface 3 of the Catalyst 1200B switch are tagged with an 802.10 header with a security association identifier of 300 as they are forwarded onto the FDDI ring. Router A can accept these packets because it is configured to route VLAN 300, classify and make a Layer 3 forwarding decision based on the destination network address and the route out (in this case Fast Ethernet 3/1), and adding the ISL VLAN header (color 200) appropriate to the destination subnet as the traffic is switched.
- There is a network requirement to bridge two VLANs together through the system rather than selectively route certain protocols. In this scenario the two VLAN IDs are placed in the same bridge group. Note that they form a single broadcast domain and spanning tree, effectively forming a single VLAN.

Figure 10 Inter-VLAN Connectivity between Multiple Switching Platforms



See the "Routing Between VLANs Configuration Example" section on page 48 for an example configuration of the topology shown in Figure 10.

To configure routing between VLANs, enter the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type slot/port.subinterface-number`	Specifies a subinterface.
Step 2	`Router(config-if)# encapsulation {sde | isl} domain`	Specifies the encapsulation type (either ISL or SDE) and the VLAN domain.
Step 3	`Router(config-if)# bridge-group bridge-group`	Associates the subinterface with the VLAN.

Configuring a Subscriber Bridge Group

The digital subscriber line (DSL) bridge support feature enables you to configure a router for intelligent bridge flooding for DSL and other bridge applications. To configure a subscriber bridge group, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# bridge bridge-group protocol {ieee | dec | vlan-bridge}`	Defines the bridge Spanning Tree Protocol.
Step 2	`Router(config)# bridge bridge-group subscriber-policy policy`	Defines a subscriber bridge group and specifies the subscriber policy for the group.
Step 3	`Router(config)# subscriber-policy policy [[no] [default] packet [permit] [deny]]`	Defines or modifies the forward and filter decisions of the subscriber policy.
Step 4	`Router(config)# interface type number`	Configures a subinterface.
Step 5	`Router(config-if)# bridge-group bridge-group [subscriber-trunk]`	Assigns a subscriber bridge group and indicates whether the interface is upstream or downstream from the traffic flow.

Note Standard access lists can coexist with the subscriber policy. However, subscriber policy will take precedence over the access list by being checked first. A packet permitted by the subscriber policy will be checked against the access list if it is specified. A packet denied by subscriber policy will be dropped with no further access list checking.

Configuring Transparent Bridging over WANs

You can configure transparent bridging over a variety of networks, as described in the following sections:

Configuring Fast-Switched Transparent Bridging over ATM

Our bridging implementation supports IEEE 802.3 frame formats and IEEE 802.10 frame formats. Our implementation can transparently bridge ARPA style Ethernet packets (also known as Ethernet version 2).

Fast-switched transparent bridging over Asynchronous Transfer Mode (ATM) supports AAL5-SNAP encapsulated packets only. All bridged AAL5-SNAP encapsulated packets are fast switched. Fast-switched transparent bridging supports Ethernet, FDDI, and Token Ring packets sent in AAL5-SNAP encapsulation over ATM. See the "Fast-Switched Transparent Bridging over ATM Example (Cisco 7000)" section on page 55 for an example configuration of fast-switched transparent bridging over ATM.

Support for RFC 1483 was added in Cisco IOS Release 12.0(3)T, enabling transparent bridging between Token Ring LANs (using AAL5-SNAP PVCs) and LANs, VLANs or ELANS (using bridged PDUs). RFC 1483 defines an encapsulation type for transferring LAN data via ATM networks.

For more information on configuring ATM, refer to the "Configuring ATM" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Configuring Transparent Bridging over DDR

The Cisco IOS software supports transparent bridging over dial-on-demand routing (DDR) and provides you some flexibility in controlling access and configuring the interface.

To configure DDR for bridging, complete the tasks in the following sections:

- Defining the Protocols to Bridge, page 14
- Specifying the Bridging Protocol, page 14
- Determining Access for Bridging, page 15
- Configuring an Interface for Bridging, page 15

For an example of configuring transparent bridging over DDR, see the "Transparent Bridging over DDR Examples" section on page 56.

Defining the Protocols to Bridge

IP packets are routed by default unless they are explicitly bridged; all others are bridged by default unless they are explicitly routed.

To bridge IP packets, use the following command in global configuration mode:

Command	Purpose
`Router(config)# no ip routing`	Disables IP routing.

If you choose *not* to bridge another protocol, use the relevant command to enable routing of that protocol. For more information about tasks and commands, refer to the relevant protocol chapters in the following publications:

- *Cisco IOS IP and IP Routing Configuration Guide*
- *Cisco IOS AppleTalk and Novell IPX Configuration Guide*
- *Cisco IOS Apollo Domain, Banyan VINES, DECnet, ISO CLNS, and XNS Configuration Guide*

Specifying the Bridging Protocol

You must specify the type of spanning-tree bridging protocol to use and also identify a bridge group. To specify the Spanning Tree Protocol and a bridge group number, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `protocol {ieee | dec | vlan-bridge}`	Defines the type of Spanning Tree Protocol and identifies a bridge group.

The bridge group number is used when you configure the interface and assign it to a bridge group. Packets are bridged only among members of the same bridge group.

Determining Access for Bridging

You can determine access by either permitting all bridge packets or by controlling access according to Ethernet type codes.

To permit all transparent bridge packets, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dialer-list` *dialer-group* **protocol bridge permit**	Defines a dialer list that permits all transparent bridge packets.

To control access by Ethernet type codes, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# access-list` *access-list-number* {**permit** | **deny**} *type-code* [*mask*]	Permits packets according to Ethernet type codes (access list numbers must be in the range 200 to 299).
Step 2	`Router(config)# dialer-list` *dialer-group* **protocol bridge list** *access-list-number*	Defines a dialer list for the specified access list.

For a table of some common Ethernet types codes, see the "Ethernet Types Codes" appendix in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Configuring an Interface for Bridging

You can configure serial interfaces or ISDN interfaces for DDR bridging. To configure an interface for DDR bridging, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface` *type number*	Specifies the serial or ISDN interface and initiates interface configuration mode.
Step 2	`Router(config-if)# dialer string` *dial-string* **dialer map bridge** [**name** *hostname*] [**broadcast**]*dial-string*[**:***isdn-subaddress*]	Configures the dial string to call. or Configures a dialer bridge map.
Step 3	`Router(config-if)# bridge-group` *bridge-group*	Assigns the specified interface to a bridge group.

Configuring Transparent Bridging over Frame Relay

The transparent bridging software supports bridging of packets over Frame Relay networks. This ability is useful for such tasks as sending packets from proprietary protocols across a Frame Relay network. Bridging over a Frame Relay network is supported both on networks that support a multicast facility and those that do not. Both cases are described in this section.

Fast-Switched Transparent Bridging

The transparent bridging software provides fast-switched transparent bridging for Frame Relay encapsulated serial and High-Speed Serial Interface (HSSI) networks.

Switched virtual circuits (SVCs) are not supported for transparent bridging in this release. All the Permanent virtual circuits (PVCs) configured on a subinterface must belong to the same bridge group.

Bridging in a Frame Relay Network with No Multicasts

The Frame Relay bridging software uses the same spanning-tree algorithm as the other bridging functions, but allows packets to be encapsulated for sending across a Frame Relay network. You specify IP-to-data-link connection identifier (DLCI) address mapping and the system maintains a table of both the Ethernet address and the DLCIs.

To configure bridging in a network that does not support a multicast facility, define the mapping between an address and the DLCI used to connect to the address. To bridge with no multicasts, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **`frame-relay map bridge`** *`dlci`* **`broadcast`**	Defines the mapping between an address and the DLCI used to connect to the address.

An example configuration is provided in the "Frame Relay Transparent Bridging Examples" section on page 53. Frame Relay is discussed in more detail in the "Configuring Frame Relay" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Bridging in a Frame Relay Network with Multicasts

The multicast facility is used to learn about the other bridges on the network, eliminating the need for you to specify any mappings with the **frame-relay map bridge broadcast** command. An example configuration is provided in the "Frame Relay Transparent Bridging Examples" section on page 53 for use as a configuration guide. Frame Relay is discussed in more detail in the "Configuring Frame Relay" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Configuring Transparent Bridging over Multiprotocol LAPB

Cisco IOS software implements transparent bridging over multiprotocol Link Access Protocol-Balanced (LAPB) encapsulation on serial interfaces. To configure transparent bridging over multiprotocol LAPB, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)#` **`interface serial`** *`number`*	Specifies the serial interface.
Step 2	`Router(config-if)#` **`no ip address`**	Specifies no IP address to the interface.
Step 3	`Router(config-if)#` **`encapsulation lapb multi`**	Configures multiprotocol LAPB encapsulation.
Step 4	`Router(config-if)#` **`bridge-group`** *`bridge-group`*	Assigns the interface to a bridge group.
Step 5	`Router(config-if)#` **`bridge`** *`bridge-group`* **`protocol`** {**`ieee`** | **`dec`** | **`vlan-bridge`**}	Specifies the type of Spanning Tree Protocol.

Note Transparent bridging over multiprotocol LAPB requires use of the **encapsulation lapb multi** command. You cannot use the **encapsulation lapb protocol** command with a **bridge** keyword to configure this feature.

For an example of configuring transparent bridging over multiprotocol LAPB, see the "Transparent Bridging over Multiprotocol LAPB Example" section on page 55".

Configuring Transparent Bridging over SMDS

We support fast-switched transparent bridging for Switched Multimegabit Data Service (SMDS) encapsulated serial and HSSI networks. Standard bridging commands are used to enable bridging on an SMDS interface.

To enable transparent bridging over SMDS, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface serial number`	Specifies the serial interface.
Step 2	`Router(config-if)# encapsulation smds`	Configures SMDS encapsulation on the serial interface.
Step 3	`Router(config-if)# bridge-group bridge-group`	Associates the interface with a bridge group.
Step 4	`Router(config-if)# smds multicast bridge smds-address`	Enables transparent bridging of packets across an SMDS network.

Broadcast Address Resolution Protocol (ARP) packets are treated differently in transparent bridging over an SMDS network than in other encapsulation methods. For SMDS, two packets are sent to the multicast address. One is sent using a standard (SMDS) ARP encapsulation; the other is sent with the ARP packet encapsulated in an 802.3 MAC header. The native ARP is sent as a regular ARP broadcast.

Our implementation of IEEE 802.6i transparent bridging for SMDS supports 802.3, 802.5, and FDDI frame formats. The router can accept frames with or without frame check sequence (FCS). Fast-switched transparent bridging is the default and is not configurable. If a packet cannot be fast switched, it is process switched.

An example configuration is provided in the "Fast-Switched Transparent Bridging over SMDS Example" section on page 56. For more information on SMDS, refer to the "Configuring SMDS" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Configuring Transparent Bridging over X.25

The transparent bridging software supports bridging of packets in X.25 frames. This ability is useful for such tasks as sending packets from proprietary protocols across an X.25 network.

The X.25 bridging software uses the same spanning-tree algorithm as the other bridging functions, but allows packets to be encapsulated in X.25 frames and sent across X.25 media. You specify the IP-to-X.121 address mapping, and the system maintains a table of both the Ethernet and X.121 addresses. To configure X.25 transparent bridging, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# x25 map bridge x.121-address broadcast [options-keywords]`	Specifies IP-to-X.121 mapping for bridging over X.25.

For more information about configuring X.25, refer to the "Configuring X.25 and LAPB" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Configuring Concurrent Routing and Bridging

You can configure the Cisco IOS software to route a given protocol among one group of interfaces and concurrently bridge that protocol among a separate group of interfaces, all within one router. The given protocol is not switched between the two groups. Rather, routed traffic is confined to the routed interfaces and bridged traffic is confined to the bridged interfaces. A protocol may be either routed or bridged on a given interface, but not both.

The concurrent routing and bridging capability is, by default, disabled. While concurrent routing and bridging is disabled, the Cisco IOS software absorbs and discards bridgeable packets in protocols that are configured for routing on any interface in the router.

When concurrent routing and bridging is first enabled in the presence of existing bridge groups, it will generate a bridge route configuration command for any protocol for which any interface in the bridge group is configured for routing. This is a precaution that applies only when concurrent routing and bridging is not already enabled, bridge groups exist, and the **bridge crb** command is encountered.

To enable concurrent routing and bridging in the Cisco IOS software, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge crb`	Enables concurrent routing and bridging.

Information about which protocols are routed and which are bridged is stored in a table, which can be displayed with the **show interfaces crb** privileged EXEC command.

When concurrent routing and bridging has been enabled, you must configure an explicit bridge route command for any protocol that is to be routed on the interfaces in a bridge group in addition to any required protocol-specific interface configuration.

To configure specific protocols to be routed in a bridge group, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge` *bridge-group* `route` *protocol*	Enables the routing of a specified protocol in a specified bridge group.

Configuring Integrated Routing and Bridging

Perform one or more of the following tasks to configure integrated routing and bridging on your router:

- Assigning a Bridge Group Number and Defining the Spanning Tree Protocol, page 8
- Configuring Interfaces, page 19
- Enabling Integrated Routing and Bridging, page 19
- Configuring the Bridge-Group Virtual Interface, page 19
- Configuring Protocols for Routing or Bridging, page 20

Assigning a Bridge Group Number and Defining the Spanning Tree Protocol

Prior to configuring the router for integrated routing and bridging, you must enable bridging by setting up a bridge group number and specifying a Spanning Tree Protocol. You can choose either the IEEE 802.1D Spanning Tree Protocol or the earlier Digital protocol upon which this IEEE standard is based.

To assign a bridge group number and define a Spanning Tree Protocol, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `protocol {ieee | dec | vlan-bridge}`	Assigns a bridge group number and defines a Spanning Tree Protocol.

The IEEE 802.1D Spanning Tree Protocol is the preferred way of running the bridge. Use the Digital Spanning Tree Protocol only for backward compatibility.

Configuring Interfaces

To configure a router interface in the Cisco IOS software, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface` *type number*	Specifies the interface and enters interface configuration mode.
Step 2	`Router(config-if)# port`	Specifies concentrator port operation.
Step 3	`Router(config-if)# bridge-group` *bridge-group*	Assigns bridge-groups to appropriate interfaces.

Enabling Integrated Routing and Bridging

After you have set up the interfaces in the router, you can enable integrated routing and bridging.

To enable integrated routing and bridging in the Cisco IOS software, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge irb`	Enables integrated routing and bridging.

Use the **show interfaces irb** privileged EXEC command to display the protocols that a given bridged interface can route to the other routed interface when the packet is routable, and to display the protocols that a given bridged interface bridges.

Configuring the Bridge-Group Virtual Interface

The bridge-group virtual interface resides in the router. It acts like a normal routed interface that does not support bridging, but represents the entire corresponding bridge group to routed interfaces within the router. The bridge-group virtual interface is assigned the number of the bridge group that it represents. The bridge-group virtual interface number is the link between the bridge-group virtual interface and its

bridge group. Because the bridge-group virtual interface is a virtual routed interface, it has all the network layer attributes, such as a network address and the ability to perform filtering. Only one bridge-group virtual interface is supported for each bridge group.

When you enable routing for a given protocol on the bridge-group virtual interface, packets coming from a routed interface but destined for a host in a bridged domain are routed to the bridge-group virtual interface, and are forwarded to the corresponding bridged interface. All traffic routed to the bridge-group virtual interface is forwarded to the corresponding bridge group as bridged traffic. All routable traffic received on a bridged interface is routed to other routed interfaces as if it is coming directly from the bridge-group virtual interface.

To create a bridge-group virtual interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# interface bvi bridge-group`	Enables a bridge-group virtual interface.

When you intend to bridge and route a given protocol in the same bridge group, you must configure the network-layer attributes of the protocol on the bridge-group virtual interface. Do not configure protocol attributes on the bridged interfaces. No bridging attributes can be configured on the bridge-group virtual interface.

Although it is generally the case that all bridged segments belonging to a bridge group are represented as a single segment or network to the routing protocol, there are situations where several individual networks coexist within the same bridged segment. To make it possible for the routed domain to learn about the other networks behind the bridge-group virtual interface, configure a secondary address on the bridge-group virtual interface to add the corresponding network to the routing process.

Configuring Protocols for Routing or Bridging

When integrated routing and bridging is enabled, the default route/bridge behavior in a bridge group is to bridge all packets.

You could then explicitly configure the bridge group to route a particular protocol, so that routable packets of this protocol are routed, while nonroutable packets of this protocol or packets for protocols for which the bridge group is not explicitly configured to route will be bridged.

You could also explicitly configure the bridge group so that it does not bridge a particular protocol, so that routable packets of this protocol are routed when the bridge is explicitly configured to route this protocol, and nonroutable packets are dropped because bridging is disabled for this protocol.

Note Packets of nonroutable protocols such as LAT are only bridged. You cannot disable bridging for the nonroutable traffic.

To configure specific protocols to be routed or bridged in a bridge group, use one or more of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# bridge bridge-group route protocol`	Enables the routing of a specified protocol in a specified bridge group.
`Router(config)# no bridge bridge-group route protocol`	Disables the routing of a specified protocol in a specified bridge group.

Command	Purpose
`Router(config)# bridge bridge-group bridge protocol`	Specifies that a protocol is to be bridged in the bridge group.
`Router(config)# no bridge bridge-group bridge protocol`	Specifies that a protocol is not to be bridged in the bridge group.

Note When a bridge group contains Token Ring interfaces, the Token Ring packets must not include RIF. The IEEE 802.1d transparent bridge standard specifies that frames with source routing information are to be dropped by transparent bridges; therefore, if Token Ring traffic includes RIF, it will be dropped. RIF is designated by the RII, which is the first bit of the MAC address. RII=1 indicates that the packet comes with RIF, RII=0 indicates that the frame does not come with RIF.

For example, to bridge AppleTalk, bridge and route IPX, and route IP in the same bridge group, you would do the following:

- Bridge AppleTalk—Because integrated routing and bridging bridges everything by default, no configuration is required to bridge AppleTalk.
- Bridge and route IPX—After using the **bridge irb** command to enable integrated routing and bridging, and the **interface bvi** command to create the bridge-group virtual interface for the bridge group, you would use the **bridge route** command to both bridge and route IPX (bridging is already enabled by default; the **bridge route** command enables routing).
- Route IP—Use the **bridge route** command to enable routing, and then use the **no bridge bridge** command to disable bridging.

Note When integrated routing and bridging is not enabled, routing a given protocol means that protocol is not bridged, and bridging a protocol means that protocol is not routed. When integrated routing and bridging is enabled, the disjunct relationship between routing and bridging is broken down, and a given protocol can be switched between routed and bridged interfaces on a selective, independent basis.

Configuring Transparent Bridging Options

You can configure one or more transparent bridging options. To configure transparent bridging options, perform one or more of the tasks in the following sections:

Disabling IP Routing

If you want to bridge IP, you must disable IP routing because IP routing is enabled by default on the Cisco IOS software. You can enable IP routing when you decide to route IP packets. To disable or enable IP routing, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# no ip routing`	Disables IP routing.
`Router(config)# ip routing`	Enables IP routing.

All interfaces in the bridge group that are bridging IP should have the same IP address. However, if you have more than one bridge group, each bridge group should have its own IP address.

Enabling Autonomous Bridging

Normally, bridging takes place on the processor card at the interrupt level. When autonomous bridging is enabled, bridging takes place entirely on the ciscoBus2 controller, significantly improving performance. Autonomous bridging is a high-speed switching feature that allows bridged traffic to be forwarded and flooded on the ciscoBus2 controller between resident interfaces. If you are using the ciscoBus2 controller, you can maximize performance by enabling autonomous bridging on the following ciscoBus2 interfaces:

- MEC
- FCIT transparent
- HSSI HDLC

Although performance improvements will be seen most in the resident interfaces, the autonomous bridging feature can also be used in bridge groups that include interfaces that are not on the ciscoBus2 controller. These interfaces include the CTR, FCI with encapsulation bridging, and HSSI with encapsulation other than HDLC, such as X.25, Frame Relay, or SMDS, MCI, STR, or SBE16.

If you enable autonomous bridging for a bridge group that includes a combination of interfaces that are resident on the ciscoBus2 controller and some that are not, the ciscoBus2 controller forwards only packets between resident interfaces. Forwarding between nonresident and resident interfaces is done in either the fast or process paths. Flooding between resident interfaces is done by the ciscoBus2 controller. Flooding between nonresident interfaces is done conventionally. If a packet is forwarded from a nonresident to a resident interface, the packet is conventionally forwarded. If packets are flooded from a nonresident interface to a resident interface, the packet is autonomously flooded.

To enable autonomous bridging on a per-interface basis, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `cbus-bridging`	Enables autonomous bridging (if using the ciscoBus2 controller).

Note You can filter by MAC-layer address on an interface only when autonomous bridging is enabled on that interface. If any filters or priority queueing is configured, autonomous bridging is automatically disabled.

Configuring LAT Compression

The local-area transport (LAT) protocol used by Digital and Digital-compatible terminal servers is one of the common protocols that lacks a well-defined network layer (Layer 3) and so always must be bridged.

To reduce the amount of bandwidth that LAT traffic consumes on serial interfaces, you can specify a LAT-specific form of compression. Doing so applies compression to LAT frames being sent out by the Cisco IOS software through the interface in question. To configure LAT compression, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `lat-compression`	Reduces the amount of bandwidth that LAT traffic consumes on a serial interface.

LAT compression can be specified only for serial interfaces. For the most common LAT operations (user keystrokes and acknowledgment packets), LAT compression reduces LAT's bandwidth requirements by nearly a factor of two.

Establishing Multiple Spanning-Tree Domains

The Cisco IEEE 802.1D bridging software supports spanning-tree domains of bridge groups. Domains are a feature specific to Cisco. This feature is only available if you have specified IEEE as the Spanning Tree Protocol. A domain establishes an external identification of the BPDUs sent from a bridge group. The purpose of this identification is as follows:

- Bridge groups defined within the domain can recognize that BPDU as belonging to them.
- Two bridged subnetworks in different domains that are sharing a common connection can use the domain identifier to identify and then ignore the BPDUs that belong to another domain. Each bridged subnetwork establishes its own spanning tree based on the BPDUs that it receives. The BPDUs it receives must contain the domain number to which the bridged subnetwork belongs. Bridged traffic is not domain identified.

Note Domains do not constrain the propagation of bridged traffic. A bridge bridges nonrouted traffic received on its interfaces regardless of domain.

You can place any number of routers or bridges within the domain. Only the devices within a domain share spanning-tree information.

When multiple routers share the same cable and you want to use only certain discrete subsets of those routers to share spanning-tree information with each other, establish spanning-tree domains. This function is most useful when running other applications, such as IP User Datagram Protocol (UDP) flooding, that use the IEEE spanning tree. You also can use this feature to reduce the number of global reconfigurations in large bridged networks.

To establish multiple spanning-tree domains, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `domain` *domain-number*	Establishes a multiple spanning-tree domain.

For an example of how to configure domains, see the "Complex Transparent Bridging Network Topology Example" section on page 57.

Preventing the Forwarding of Dynamically Determined Stations

Normally, the system forwards any frames for stations that it has learned about dynamically. By disabling this activity, the bridge will only forward frames whose address have been statically configured into the forwarding cache. To prevent or allow forwarding of dynamically determined stations, use one of the following command in global configuration mode:

Command	Purpose
`Router(config)# no bridge bridge-group acquire`	Filters out all frames except those whose addresses have been statically configured into the forwarding cache.
`Router(config)# bridge bridge-group acquire`	Forwards any frames for stations that the system has learned about dynamically.

Forwarding Multicast Addresses

A packet with a RIF, indicated by a source address with the multicast bit turned on, is not usually forwarded. However, you can configure bridging support to allow the forwarding of frames that would otherwise be discarded because they have a RIF. Although you can forward these frames, the bridge table will not be updated to include the source addresses of these frames.

To forward frames with multicast addresses, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group multicast-source`	Allows the forwarding of frames with multicast source addresses.

Configuring Bridge Table Aging Time

A bridge forwards, floods, or drops packets based on the bridge table. The bridge table maintains both static entries and dynamic entries. Static entries are entered by the network manager or by the bridge itself. Dynamic entries are entered by the bridge learning process. A dynamic entry is automatically removed after a specified length of time, known as *aging time*, from the time the entry was created or last updated.

If hosts on a bridged network are likely to move, decrease the aging-time to enable the bridge to adapt to the change quickly. If hosts do not send continuously, increase the aging time to record the dynamic entries for a longer time and thus reduce the possibility of flooding when the hosts send again.

To set the aging time, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge-group bridge-group aging-time seconds`	Sets the bridge table aging time.

Filtering Transparently Bridged Packets

A bridge examines frames and sends them through the internetwork according to the destination address; a bridge will not forward a frame back to its originating network segment. The bridge software allows you to configure specific administrative filters that filter frames based upon information other than paths to their destinations. You can perform administrative filtering by performing one of the tasks in the following sections:

- Setting Filters at the MAC Layer, page 25
- Filtering LAT Service Announcements, page 29

Note When setting up administrative filtering, remember that there is virtually no performance penalty in filtering by Media Access Control (MAC) address or vendor code, but there can be a significant performance penalty when filtering by protocol type.

When configuring transparent bridging access control, keep the following points in mind:

- You can assign only one access list to an interface.
- The conditions in the access list are applied to all outgoing packets not sourced by the Cisco IOS software.
- Access lists are scanned in the order you enter them; the first match is used.
- An implicit deny everything entry is automatically defined at the end of an access list unless you include an explicit permit everything entry at the end of the list.
- All new entries to an existing list are placed at the end of the list. You cannot add an entry to the middle of a list. This means that if you have previously included an explicit permit everything entry, new entries will never be scanned. The solution is to delete the access list and retype it with the new entries.
- You can create extended access lists to specify more detailed filters, such as address match only.
- You should not use extended access lists on FDDI interfaces doing transit bridging as opposed to translational bridging.
- Configuring bridging access lists of type 700 may cause a momentary interruption of traffic flow.

For more information on access lists, refer to the "Traffic Filtering and Firewalls" chapter of the *Cisco IOS Security Configuration Guide*.

Setting Filters at the MAC Layer

You can filter the sending of frames at the MAC layer by performing tasks in one of the following sections:

- Filtering by Specific MAC Address, page 26
- Filtering by Vendor Code, page 26
- Filtering by Protocol Type, page 27

When filtering by a MAC-layer address, you can use two kinds of access lists: standard access lists that specify a simple address, and extended access lists that specify two addresses. You can also further restrict access by creating filters for these lists. After you have completed one of the preceding tasks, perform the task in the "Defining and Applying Extended Access Lists" section on page 28.

Note MAC addresses on Ethernets are "bit swapped" when compared with MAC addresses on Token Ring and FDDI. For example, address 0110.2222.3333 on Ethernet is 8008.4444.CCCC on Token Ring and FDDI. Access lists always use the canonical Ethernet representation. When using different media and building access lists to filter on MAC addresses, keep this point in mind. Note that when a bridged packet traverses a serial link, it has an Ethernet-style address.

Filtering by Specific MAC Address

You can filter frames with a particular MAC-layer station source or destination address. Any number of addresses can be configured into the system without a performance penalty. To filter by MAC-layer address, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group address mac-address {forward | discard}[interface]`	Filters particular MAC-layer station addresses.

When filtering specific MAC destination addresses, allow for multicast or broadcast packets that are required by the bridged network protocols. Refer to the example in the "Multicast or Broadcast Packets Bridging Example" section on page 51 to guide you in building your configuration to allow for multicast or broadcast packets.

Filtering by Vendor Code

The bridging software allows you to create access lists to administratively filter MAC addresses. These access lists can filter groups of MAC addresses, including those with particular vendor codes. There is no noticeable performance loss in using these access lists, and the lists can be of indefinite length. You can filter groups of MAC addresses with particular vendor codes by performing the first task and one or both of the other tasks that follow:

- Establish a vendor code access list
- Filter source addresses
- Filter destination addresses

To establish a vendor code access list, use the following command in global configuration mode:

Command	Purpose
`Router(config)# access-list access-list-number {permit | deny} address mask`	Prepares access control information for filtering of frames by canonical (Ethernet-ordered) MAC address.

The vendor code is the first three bytes of the MAC address (left to right). For an example of how to filter by vendor code, see the "Multicast or Broadcast Packets Bridging Example" section on page 51.

Note Remember that, as with any access list using MAC addresses, Ethernets swap their MAC address bit ordering, and Token Rings and FDDI do not. Therefore, an access list that works for one medium might not work for others.

Once you have defined an access list to filter by a particular vendor code, you can assign an access list to a particular interface for filtering on the MAC *source* addresses of packets *received* on that interface or the MAC *destination* addresses of packets that would ordinarily be *forwarded* out that interface. To filter by source or destination addresses, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* **input-address-list** *access-list-number*	Assigns an access list to a particular interface.
`Router(config-if)# bridge-group` *bridge-group* **output-address-list** *access-list-number*	Assigns an access list to an interface for filtering by the MAC destination addresses.

Filtering by Protocol Type

You can filter by protocol type by using the access-list mechanism and specifying a protocol type code. To filter by protocol type, perform the first task and one or more of the other tasks that follow:

- Establish a protocol type access list
- Filter Ethernet- and SNAP-encapsulated packets on input
- Filter Ethernet- and SNAP-encapsulated packets on output
- Filter IEEE 802.2-encapsulated packets on input
- Filter IEEE 802.2-encapsulated packets on output

Note It is a good idea to have both input and output type code filtering on different interfaces.

The order in which you enter **access-list** commands affects the order in which the access conditions are checked. Each condition is tested in succession. A matching condition is then used to execute a permit or deny decision. If no conditions match, a "deny" decision is reached.

Note Protocol type access lists can have an impact on system performance; therefore, keep the lists as short as possible and use wildcard bit masks whenever possible.

Access lists for Ethernet- and IEEE 802.2-encapsulated packets affect only bridging functions. It is not possible to use such access lists to block frames with protocols that are being routed.

You can establish protocol type access lists. Specify either an Ethernet type code for Ethernet-encapsulated packets or a DSAP/SSAP pair for 802.3 or 802.5-encapsulated packets. Ethernet type codes are listed in the "Ethernet Type Codes" appendix of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

To establish protocol type access lists, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **access-list** *access-list-number* {**permit** | **deny**} *type-code wild-mask*	Prepares access control information for filtering frames by protocol type.

You can filter Ethernet- and SNAP-encapsulated packets on input. For SNAP-encapsulated frames, the access list you create is applied against the two-byte TYPE field given after the DSAP/SSAP/OUI fields in the frame. The access list is applied to all Ethernet and SNAP frames received on that interface prior to the bridge learning process. SNAP frames also must pass any applicable IEEE 802.2 DSAP/SSAP access lists.

You can also filter Ethernet- and SNAP-encapsulated packets on output. The access list you create is applied just before sending out a frame to an interface.

To filter these packets on input or output, use the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* **input-type-list** *access-list-number*	Adds a filter for Ethernet- and SNAP-encapsulated packets on input.
`Router(config-if)# bridge-group` *bridge-group* **output-type-list** *access-list-number*	Adds a filter for Ethernet- and SNAP-encapsulated packets on output.

You can filter IEEE 802-encapsulated packets on input. The access list you create is applied to all IEEE 802 frames received on that interface prior to the bridge-learning process. SNAP frames also must pass any applicable Ethernet type-code access list.

You can also filter IEEE 802-encapsulated packets on output. SNAP frames also must pass any applicable Ethernet type-code access list. The access list you create is applied just before sending out a frame to an interface.

To filter these packets on input or output, use the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* **input-lsap-list** *access-list-number*	Adds a filter for IEEE 802-encapsulated packets on input.
`Router(config-if)# bridge-group` *bridge-group* **output-lsap-list** *access-list-number*	Adds a filter for IEEE 802-encapsulated packets on output.

Access lists for Ethernet- and IEEE 802-encapsulated packets affect only bridging functions. You cannot use such access lists to block frames with protocols that are being routed.

Defining and Applying Extended Access Lists

If you are filtering by the MAC-layer address, whether it is by a specific MAC address, vendor code, or protocol type, you can define and apply extended access lists. Extended access lists allow finer granularity of control. They allow you to specify both source and destination addresses and arbitrary bytes in the packet.

To define an extended access list, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **access-list** *access-list-number* {**permit** | **deny**} *source source-mask destination destination-mask offset size operator operand*	Defines an extended access list for finer control of bridged traffic.

To apply an extended access list to an interface, use the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `input-pattern-list` *access-list-number*	Applies an extended access list to the packets being received by an interface.
`Router(config-if)# bridge-group` *bridge-group* `output-pattern-list` *access-list-number*	Applies an extended access list to the packet being sent by an interface.

After an access list is created initially, any subsequent additions (possibly entered from the terminal) are placed at the *end* of the list. In other words, you cannot selectively add or remove access list command lines from a specific access list.

Caution Because of their complexity, only use extended access lists if you are very familiar with the Cisco IOS software. Further, do not specify an offset value that is greater than the size of the packet.

Filtering LAT Service Announcements

The bridging software allows you to filter LAT frames. LAT bridge filtering allows the selective inclusion or exclusion of LAT multicast service announcements on a per-interface basis.

Note The LAT filtering commands are not implemented for Token Ring interfaces.

In the LAT protocol, a *group code* is defined as a decimal number in the range 0 to 255. Some of the LAT configuration commands take a list of group codes; this is referred to as a *group code list*. The rules for entering numbers in a group code list follow:

- Entries can be individual group code numbers separated with a space. (The Digital LAT implementation specifies that a list of numbers be separated by commas; however, our implementation expects the numbers to be separated by spaces.)
- Entries can also specify a range of numbers. This is done by separating an ascending order range of group numbers with hyphens.
- Any number of group codes or group code ranges can be listed in one command; just separate each with a space.

In LAT, each node sends a periodic service advertisement message that announces its existence and availability for connections. Within the message is a group code list; this is a mask of up to 256 bits. Each bit represents a group number. In the traditional use of LAT group codes, a terminal server only will connect to a host system when there is an overlap between the group code list of the user on the terminal server and the group code list in the service advertisement message. In an environment with many bridges and many LAT hosts, the number of multicast messages that each system has to deal with becomes unreasonable. The 256 group codes might not be enough to allocate local assignment policies, such as giving each DECserver 200 device its own group code in large bridged networks. LAT group code filtering allows you to have very fine control over which multicast messages actually get bridged. Through a combination of input and output permit and deny lists, you can implement many different LAT control policies.

You can filter LAT service advertisements by performing any of the tasks in the following sections:

- Enabling LAT Group Code Service Filtering, page 30
- Specifying Deny or Permit Conditions for LAT Group Codes on Input, page 30
- Specifying Deny or Permit Conditions for LAT Group Codes on Output, page 30

Enabling LAT Group Code Service Filtering

You can specify LAT group-code filtering to inform the system that LAT service advertisements require special processing. To enable LAT group-code filtering, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group lat-service-filtering`	Enables LAT service filtering.

Specifying Deny or Permit Conditions for LAT Group Codes on Input

You can specify the group codes by which to deny or permit access upon input. Specifying deny conditions causes the system to not bridge any LAT service advertisement that contain any of the specified groups. Specifying permit conditions causes the system to bridge only those service advertisements that match at least one group in the specified group list.

To specify deny or permit conditions for LAT groups on input, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group bridge-group input-lat-service-deny group-list`	Specifies the group codes with which to deny access upon input.
`Router(config-if)# bridge-group bridge-group input-lat-service-permit group-list`	Specifies the group codes with which to permit access upon input.

If a message specifies group codes in both the deny and permit list, the message is not bridged.

Specifying Deny or Permit Conditions for LAT Group Codes on Output

You can specify the group codes by which to deny or permit access upon output. Specifying deny conditions causes the system to not bridge onto the output interface any LAT service advertisements that contain any of the specified groups. Specifying permit conditions causes the system to bridge onto the output interface only those service advertisements that match at least one group in the specified group list.

To specify deny or permit conditions for LAT groups on output, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bridge-group bridge-group output-lat-service-deny group-list`	Specifies the group codes with which to deny access upon output.
`Router(config-if)# bridge-group bridge-group output-lat-service-permit group-list`	Specifies the group codes with which to permit access upon output.

If a message matches both a deny and a permit condition, it will not be bridged.

Adjusting Spanning-Tree Parameters

You might need to adjust certain spanning-tree parameters if the default values are not suitable for your bridge configuration. Parameters affecting the entire spanning tree are configured with variations of the **bridge** global configuration command. Interface-specific parameters are configured with variations of the **bridge-group** interface configuration command.

You can adjust spanning-tree parameters by performing any of the tasks in the following sections:

- Setting the Bridge Priority, page 31
- Setting an Interface Priority, page 31
- Assigning Path Costs, page 32
- Adjusting BPDU Intervals, page 32
- Disabling the Spanning Tree on an Interface, page 33

Note Only network administrators with a good understanding of how bridges and the Spanning Tree Protocol work should make adjustments to spanning-tree parameters. Poorly planned adjustments to these parameters can have a negative impact on performance. A good source on bridging is the IEEE 802.1D specification; see the "References and Recommended Reading" appendix in the *Cisco IOS Configuration Fundamentals Command Reference* for other references.

Setting the Bridge Priority

You can globally configure the priority of an individual bridge when two bridges tie for position as the root bridge, or you can configure the likelihood that a bridge will be selected as the root bridge. This priority is determined by default; however, you can change it. To set the bridge priority, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `priority` *number*	Sets the bridge priority.

Setting an Interface Priority

You can set a priority for an interface. When two bridges tie for position as the root bridge, you configure an interface priority to break the tie. The bridge with the lowest interface value is elected. To set an interface priority, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `priority` *number*	Establishes a priority for a specified interface.

Assigning Path Costs

Each interface has a path cost associated with it. By convention, the path cost is 1000/data rate of the attached LAN, in millions of bits per second (Mbps). You can set different path costs. Refer to the entry for this command in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2) for the various media defaults. To assign path costs, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `path-cost` *cost*	Sets a path cost different from the defaults.

Adjusting BPDU Intervals

You can adjust BPDU intervals as described in the following sections:

- Adjusting the Interval between Hello BPDUs, page 32
- Defining the Forward Delay Interval, page 32
- Defining the Maximum Idle Interval, page 33

Note Each bridge in a spanning tree adopts the interval between hello BPDUs, the forward delay interval, and the maximum idle interval parameters of the root bridge, regardless of what its individual configuration might be.

Adjusting the Interval between Hello BPDUs

You can specify the interval between hello BPDUs. To adjust this interval, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `hello-time` *seconds*	Specifies the interval between hello BPDUs.

Defining the Forward Delay Interval

The forward delay interval is the amount of time spent listening for topology change information after an interface has been activated for bridging and before forwarding actually begins. To change the default interval setting, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `forward-time` *seconds*	Sets the default of the forward delay interval.

Defining the Maximum Idle Interval

If a bridge does not hear BPDUs from the root bridge within a specified interval, it assumes that the network has changed and recomputes the spanning-tree topology. To change the default interval setting, using the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `max-age` *seconds*	Changes the amount of time a bridge will wait to hear BPDUs from the root bridge.

Disabling the Spanning Tree on an Interface

When a *loop-free* path exists between any two bridged subnetworks, you can prevent BPDUs generated in one transparent bridging subnetwork from impacting nodes in the other transparent bridging subnetwork, yet still permit bridging throughout the bridged network as a whole. For example, when transparently bridged LAN subnetworks are separated by a WAN, BPDUs can be prevented from traveling across the WAN link.

To disable the spanning tree on an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group` *bridge-group* `spanning-disabled`	Disables the spanning tree on an interface.

Configuring Transparent and IRB Bridging on a PA-12E/2FE Ethernet Switch

The PA-12E/2FE Ethernet switch port adapter provides Cisco 7200 series routers with up to twelve 10-Mbps and two 10/100-Mbps switched Ethernet (10BASE-T) and Fast Ethernet (100BASE-TX) interfaces for an aggregate bandwidth of 435 Mbps, full-duplex. The PA-12E/2FE port adapter supports the Ethernet, IEEE 802.3, and IEEE 802.3u specifications for 10-Mbps and 100-Mbps transmission over UTP cables.

The PA-12E/2FE port adapter offloads Layer 2 switching from the host CPU by using store-and-forward or cut-through switching technology between interfaces within the same VLAN on the PA-12E/2FE port adapter. The PA-12E/2FE port adapter supports up to four VLANs (bridge groups).

Note The PA-12E/2FE port adapter is a dual-width port adapter, which means it occupies two horizontally aligned port adapter slots when installed in a Cisco 7200 series router. (Single-width port adapters occupy individual port adapter slots in a Cisco 7200 series router.)

All interfaces on the PA-12E/2FE port adapter support autosensing and autonegotiation of the proper transmission mode (half-duplex or full-duplex) with an attached device. The first two PA-12E/2FE interfaces (port 0 and port 1) also support autosensing and autonegotiation of the proper connection speed (10 Mbps or 100 Mbps) with an attached device. If an attached device does not support autosensing and autonegotiation of the proper transmission mode, the PA-12E/2FE interfaces attached to the device automatically enter half-duplex mode. Use the **show running-config** command to determine if a PA-12E/2FE interface is autosensing and autonegotiating the proper transmission mode with an attached device. Use the **full-duplex** and the **half-duplex** commands to change the transmission mode of a PA-12E/2FE interface. After changing the transmission mode, use the **show interfaces** command to verify the interface's transmission mode.

Note If you use the **full-duplex** and the **half-duplex** commands to change the transmission mode of the first two PA-12E/2FE interfaces (port 0 and port 1), the transmission speed of the two PA-12E/2FE interfaces automatically defaults to 100-Mbps. The first two PA-12E/2FE interfaces only operate at 10-Mbps when the interfaces are autosensing and autonegotiating the proper connection speed (10-Mbps or 100-Mbps) with an attached device.

To configure the PA-2E/2FE port adapter, perform the tasks in the following sections. The first task is required, all other tasks are optional.

- Configuring the PA-12E/2FE Port Adapter, page 34
- Monitoring and Maintaining the PA-12E/2FE Port Adapter, page 36
- Configuring Bridge Groups Using the 12E/2FE VLAN Configuration WebTool, page 36

Note If you plan to use a PA-12E/2FE interface to boot from a network (TFTP), ensure that the interface is configured for a loop-free environment, an IP address is configured for the interface's bridge-group virtual interface, and system boot image in release 11.2(10)P is installed on your router (use the **show version** command to view your router's system boot image). Then, *before* booting from the network server, use the **bridge-group** *bridge-group number* **spanning-disabled** command to disable the Spanning Tree Protocol configured on the interface to keep the TFTP server from timing out and closing the session.

For detailed information about boot from a network (TFTP), loading a system image from a network server, and configuring the Spanning Tree Protocol on your Cisco 7200 series router, refer to the *PA-12E/2FE Ethernet Switch10BASE-T and 100BASE-TX Port Adapter Installation and Configuration* that accompanies the hardware and to the *Cisco IOS Configuration Fundamentals Configuration Guide* and *Cisco IOS Bridging and IBM Networking Configuration Guide* publications.

For information on other commands that can be used to configure a PA-12E/2FE port adapter, refer to the "Configuring Interfaces" chapter in the *Cisco IOS Configuration Fundamentals Configuration Guide*.

For PA-2E/2FE port adapter configuration examples, see the "Configuration of Transparent Bridging for PA-12E/2FE Port Adapter Example" section on page 62 and the "Configuration of IRB for PA-12E/2FE Port Adapter Example" section on page 62.

Configuring the PA-12E/2FE Port Adapter

This section provides instructions for a basic configuration. You might also need to enter other configuration commands depending on the requirements for your system configuration and the protocols you plan to route on the interface.

To configure the interfaces on the PA-12E/2FE port adapter, perform the following tasks beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# bridge bridge-group protocol ieee`	Specifies the type of Spanning Tree Protocol. The PA-12E/2FE port adapter supports DEC and IEEE Spanning Tree Protocols; however, we recommend using the IEEE protocol when configuring bridge groups.
Step 2	`Router(config)# interface fastethernet slot/port` (for ports 0 and 1) `interface ethernet slot/port` (for ports 2 through 13)	Enters the interface you want to configure.
Step 3	`Router(config-if)# bridge-group bridge-group`	Assigns a bridge group to the interface.
Step 4	`Router(config-if)# cut-through [receive | transmit]`	(Optional) configures the interface for cut-through switching technology. The default is store-and-forward.
Step 5	`Router(config-if)# full-duplex`	(Optional) if an attached device does not support autosensing or autonegotiation, configures the transmission mode for full-duplex. The default is half-duplex.
Step 6	`Router(config-if)# no shutdown`	Changes the shutdown state to up.
Step 7	`Router(config-if)# exit`	Returns to global configuration mode.
Step 8		Repeat Step 1 through Step 7 for each interface.
Step 9	`Router(config)# exit`	Exits global configuration mode.
Step 10	`Router# copy running-config startup-config`	Saves the new configuration to memory.

To enable integrated routing and bridging on the bridge groups, perform the following tasks beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# bridge irb`	Enables integrated routing and bridging.
Step 2	`Router(config)# interface bvi bridge-group`	Enables a virtual interface on a bridge group.
Step 3	`Router(config-if)# ip address address mask`	Assigns an IP address and subnet mask to the bridge group virtual interface.
Step 4	`Router(config-if)# no shutdown`	Changes the shutdown state to up.
Step 5	`Router(config-if)# exit`	Returns to global configuration mode.
Step 6		Repeat Step 1 through Step 5 for each interface.
Step 7	`Router(config)# bridge bridge-group route protocol`	Specifies the protocol for each bridge group.
Step 8	`Router(config)# exit`	Exits global configuration mode.
Step 9	`Router# copy running-config startup-config`	Saves the new configuration to memory.

Monitoring and Maintaining the PA-12E/2FE Port Adapter

After configuring the new interface, you can display its status and verify other information. To display information about the PA-12E/2FE port adapter, perform the following tasks in privileged EXEC mode:

Command	Purpose
`Router#` **`show version`**	Displays the configuration of the system hardware, the software version, the names and sources of configuration files, and the boot image.
`Router#` **`show controllers`**	Displays all current port adapters and their interfaces.
`Router#` **`show interface fastethernet`** *`slot/port`* (for ports 0 and 1) **`show interface ethernet`** *`slot/port`* (for ports 2 through 13)	Verifies the interfaces have the correct slot number and that the interface and line protocol are in the correct state.
`Router#` **`show bridge group`**	Verifies all bridge groups and their interfaces.
`Router#` **`show interface ethernet`** *`slot/port`* **`irb`** (for ports 2 through 13) **`show interface fastethernet`** *`slot/port`* **`irb`** (for ports 0 and 1)	Verifies the correct routed protocol is configured for each interface.
`Router#` **`show protocols`**	Displays the protocols configured for the entire system and specific interfaces.
`Router#` **`show pas eswitch addresses fastethernet`** *`slot/port`* (for ports 0 and 1) **`show pas eswitch addresses ethernet`** *`slot/port`* (for ports 2 through 13)	Displays the Layer 2 learned addresses for each interface.
`Router#` **`show running-config`**	Displays the running configuration file.
`Router#` **`show startup-config`**	Displays the configuration stored in NVRAM.

Configuring Bridge Groups Using the 12E/2FE VLAN Configuration WebTool

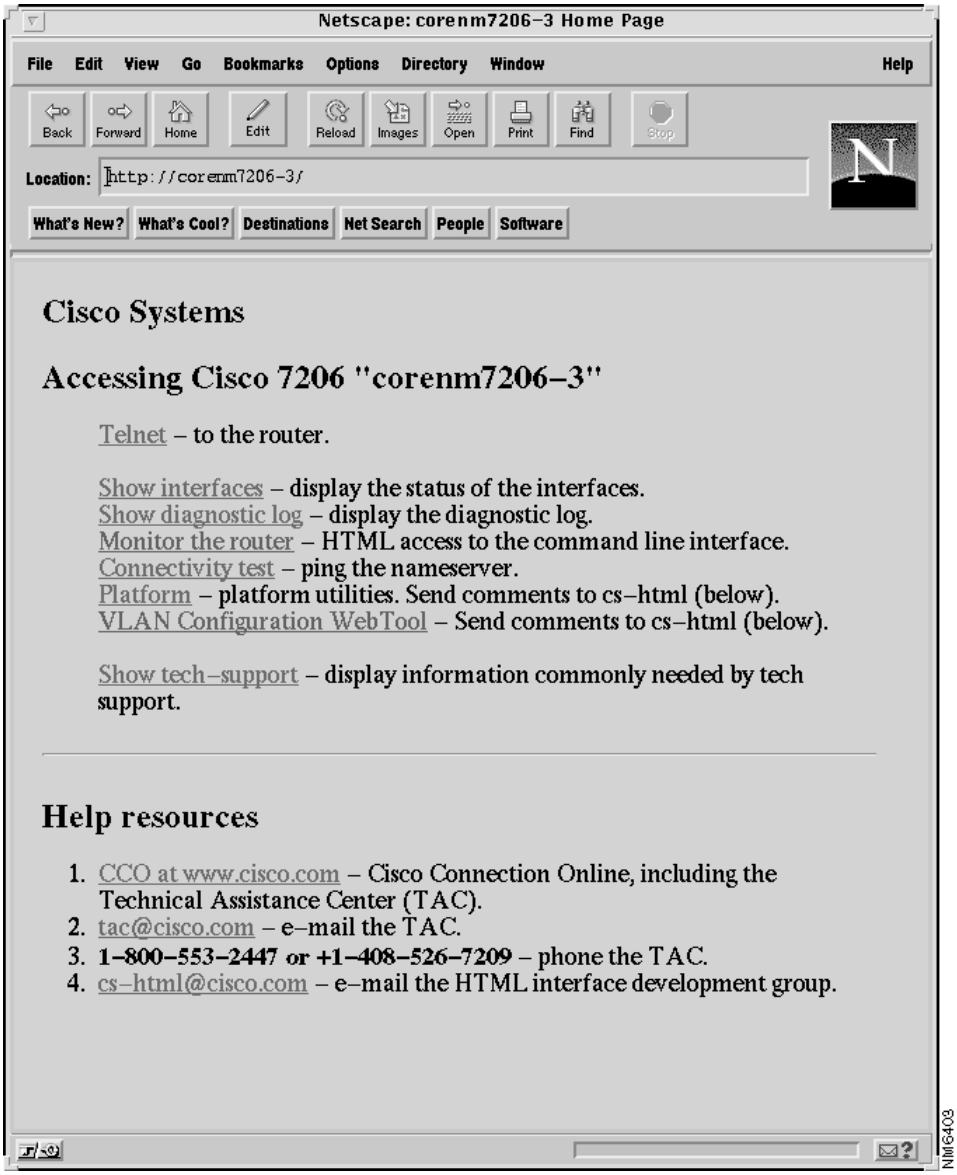
The 12E/2FE VLAN Configuration WebTool, shown in Figure 11, is a Web browser-based Java applet that displays configured interfaces and bridge groups for PA-12E/2FE port adapters installed in Cisco routers. With the WebTool you can perform the following tasks:

- Create and delete bridge groups (also referred to as VLANs)
- Add and remove PA-12E/2FE interfaces from bridge groups
- Assign colors to bridge groups and PA-12E/2FE interfaces
- Administratively shut down (disable) and bring up (enable) PA-12E/2FE interfaces
- View the bridge-group status of each PA-12E/2FE interface

You can access the 12E/2FE VLAN Configuration WebTool from your router's home page. For more information on the router's home page, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide*.

For complete procedures on how to use the VLAN Configuration WebTool, refer to the *PA-12E/2FE Ethernet Switch10BASE-T and 100BASE-TX Port Adapter Installation and Configuration* that accompanies the hardware.

Figure 11 Example Home Page for a Cisco 7200 Series Router (Cisco 7206 Shown)



Note You must use a Java enabled Web browser to access the 12E/2FE VLAN Configuration WebTool from your router's home page.

All Cisco routers running Cisco IOS Release 11.0 or later have a home page. If your router has an installed PA-12E/2FE port adapter, you can access the 12E/2FE VLAN Configuration WebTool from the router's home page.

Note All Cisco router home pages are password protected. Contact your network administrator if you do not have the name or password for your Cisco 7200 series router.

Note The VLAN Configuration WebTool hypertext link is listed in the router's home page *only* when a PA-12E/2FE port adapter is installed in the router.

Tuning the Transparently Bridged Network

The following sections describe how to configure features that enhance network performance by reducing the number of packets that traverse the backbone network:

- Configuring Circuit Groups, page 38
- Configuring Constrained Multicast Flooding, page 39

Configuring Circuit Groups

In the process of loop elimination, the spanning-tree algorithm always blocks all but one of a group of parallel network segments between two bridges. When those segments are of limited bandwidth, it might be preferable to augment the aggregate bandwidth between two bridges by forwarding across multiple parallel network segments. Circuit groups can be used to group multiple parallel network segments between two bridges to distribute the load while still maintaining a loop-free spanning tree.

Deterministic load distribution distributes traffic between two bridges across multiple parallel network segments grouped together into a single circuit group. As long as one port of the circuit group is in the forwarding state, all ports in that circuit group will participate in load distribution regardless of their spanning-tree port states. This process guarantees that the computed spanning tree is still adaptive to any topology change and the load is distributed among the multiple segments. Deterministic load distribution guarantees packet ordering between source-destination pairs, and always forwards traffic for a source-destination pair on the same segment in a circuit group for a given circuit-group configuration.

Note You should configure all parallel network segments between two bridges into a single circuit group. Deterministic load distribution across a circuit group adjusts dynamically to the addition or deletion of network segments, and to interface state changes.

If a circuit-group port goes down and up as a result of configuration or a line protocol change, the spanning-tree algorithm will bypass port transition and will time out necessary timers to force the eligible circuit-group ports to enter the forwarding state. This avoids the long disruption time caused by spanning-tree topology recomputation and therefore resumes the load distribution as quickly as possible.

To tune the transparently bridged network, perform the following tasks:

1. Define a circuit group.
2. Optionally, configure a transmission pause interval.
3. Modify the load distribution strategy.

To define a circuit group, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bridge-group bridge-group circuit-group circuit-group`	Adds a serial interface to a circuit group.

For circuit groups of mixed-bandwidth serial interfaces, it might be necessary to configure a pause interval during which the sending of data is suspended to avoid ordering packets incorrectly following changes in the composition of a circuit group. Changes in the composition of a circuit group include the addition or deletion of an interface and interface state changes. To configure a transmission pause interval, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group circuit-group circuit-group pause milliseconds`	Configures a transmission pause interval.

For applications that depend on the ordering of mixed unicast and multicast traffic from a given source, load distribution must be based upon the source MAC address only. To modify the load distribution strategy to accommodate such applications, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group circuit-group circuit-group source-based`	Distributes base load on the source MAC address only.

For an example of how to configure a circuit group, see the "Complex Transparent Bridging Network Topology Example" section later in this chapter.

Configuring Constrained Multicast Flooding

In a transparent bridge, multicast packets are flooded on all forwarding ports on the bridge. For some protocols, it is possible for a bridge to determine the membership of multicast groups, and constrain the flooding of multicasts to a subset of the forwarding ports. Constrained multicast flooding enables a bridge to determine group membership of IP multicast groups dynamically and flood multicast packets only on those ports that reach group members.

To enable constrained multicast flooding, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge cmf`	Enables constrained multicast flooding for all configured bridge groups.

Monitoring and Maintaining the Transparent Bridge Network

To monitor and maintain activity on the bridged network, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# clear bridge bridge-group`	Removes any learned entries from the forwarding database and clears the transmit and receive counts for any statically configured forwarding entries.
`Router# clear bridge [bridge-group] multicast [router-ports | groups | counts] [group-address] [interface-unit] [counts]`	Removes multicast-group state information and clears the transmit and receive counts.
`Router# clear sse`	Reinitializes the Silicon Switch Processor (SSP) on the Cisco 7000 series router.
`Router# clear vlan statistics`	Removes VLAN statistics from any statically or system-configured entries.
`Router# show bridge [bridge-group] [interface]`	Displays details of the bridge group.
`Router# show bridge [bridge-group] [interface] [address [mask]] [verbose]`	Displays classes of entries in the bridge forwarding database.
`Router# show bridge [bridge-group] circuit-group [circuit-group] [src-mac-address] [dst-mac-address]`	Displays the interfaces configured in each circuit group and shows whether they are participating in load distribution.
`Router# show bridge [bridge-group] multicast [router-ports | groups] [group-address]`	Displays transparent bridging multicast state information.
`Router# show bridge group [verbose]`	Displays information about configured bridge groups.
`Router# show bridge vlan`	Displays IEEE 802.10 transparently bridged VLAN configuration.
`Router# show interfaces crb`	Displays the configuration for each interface that has been configured for routing or bridging.
`Router# show interfaces [interface] irb`	Displays the protocols that can be routed or bridged for the specified interface.
`Router# show span`	Displays the spanning-tree topology known to the router, including whether or not filtering is in effect.
`Router# show sse summary`	Displays a summary of SSP statistics.
`Router# show subscriber-policy policy`	Displays the details of a subscriber policy.
`Router# show vlans`	Displays VLAN subinterfaces.

Transparent and SRT Bridging Configuration Examples

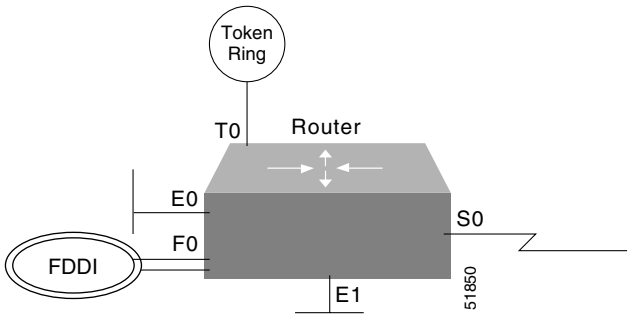
The following sections provide example configurations that you can use as a guide to configuring your bridging environment:

- Basic Bridging Example, page 41
- Concurrent Routing and Bridging Example, page 42
- Basic Integrated Routing and Bridging Example, page 43
- Complex Integrated Routing and Bridging Example, page 43

Basic Bridging Example

Figure 12 is an example of a basic bridging configuration. The system has two Ethernets, one Token Ring, one FDDI port, and one serial line. IP traffic is routed and everything else is bridged. The Digital-compatible bridging algorithm with default parameters is being used.

Figure 12 Example of Basic Bridging



The configuration file for the router in Figure 12 is as follows:

```
interface tokenring 0
 ip address 131.108.1.1 255.255.255.0
 bridge-group 1
!
interface fddi 0
 ip address 131.108.2.1 255.255.255.0
 bridge-group 1
```

```
!
interface ethernet 0
 ip address 192.31.7.26 255.255.255.240
 bridge-group 1
!
interface serial 0
 ip address 192.31.7.34 255.255.255.240
 bridge-group 1
!
interface ethernet 1
 ip address 192.31.7.65 255.255.255.240
 bridge-group 1
!
bridge 1 protocol dec
```

Concurrent Routing and Bridging Example

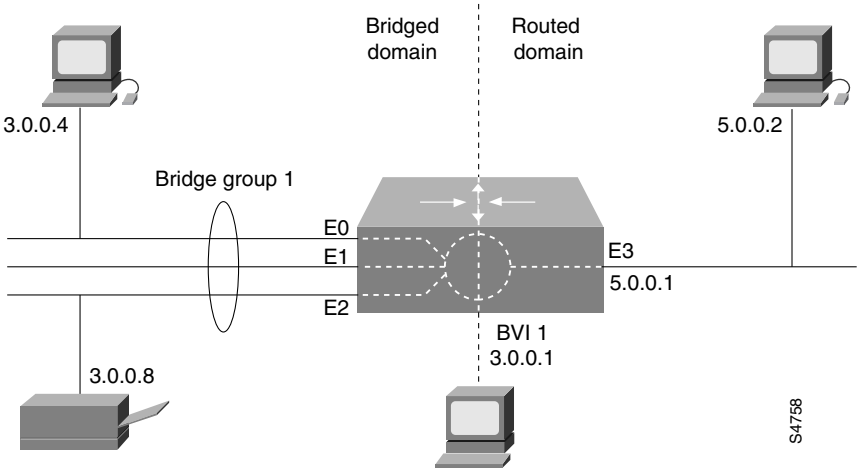
In the following example DECnet and IPX are concurrently routed and bridged. IP and AppleTalk are routed on all interfaces, DECnet and IP are routed on all interfaces not in the bridge group, and all protocols other than IP and AppleTalk are bridged on all interfaces in the bridge group:

```
!
ipx routing 0000.0c36.7a43
appletalk routing
!
decnet routing 9.65
decnet node-type routing-iv
!
interface Ethernet0/0
 ip address 172.19.160.65 255.255.255.0
 ipx network 160
 appletalk address 160.65
 decnet cost 7
!
interface Ethernet0/1
 ip address 172.19.161.65 255.255.255.0
 ipx network 161
 appletalk address 161.65
 decnet cost 7
!
interface Ethernet0/2
 ip address 172.19.162.65 255.255.255.0
 appletalk address 162.65
 bridge-group 1
!
interface Ethernet0/3
 ip address 172.19.14.65 255.255.255.0
 appletalk address 14.65
 appletalk zone california
 bridge-group 1
!
router igrp 666
network 172.19.0.0


!
bridge crb
bridge 1 protocol ieee
bridge 1 route appletalk
bridge 1 route ip
!
```

Basic Integrated Routing and Bridging Example

Figure 13 is an example of integrated routing and bridging that uses Bridge-Group 1 to bridge and route IP. The router has three bridged Ethernet interfaces and one routed Ethernet interface.

Figure 13 Basic IP Routing using Integrated Routing and Bridging



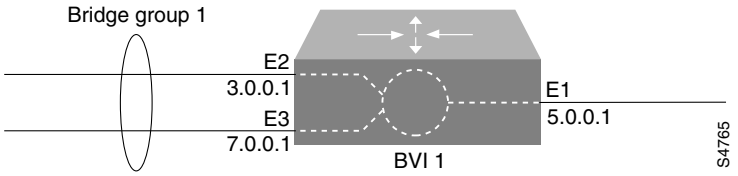
The following example shows the relevant portions of the configuration for the router in Figure 13:

```
interface Ethernet 0
 bridge-group 1
!
interface Ethernet 1
 bridge-group 1
!
interface Ethernet 2
 bridge-group 1
!
interface Ethernet 3
 ip address 5.0.0.1 255.0.0.0
!
interface BVI 1
 ip address 3.0.0.1 255.0.0.0
!
bridge irb
bridge 1 protocol ieee
 bridge 1 route ip
```

Complex Integrated Routing and Bridging Example

Figure 14 is a more complex example of integrated routing and bridging, where bridge group 1 is used to route IP traffic, bridge IPX traffic, and bridge and route AppleTalk traffic.

Figure 14 Complex Integrated Routing and Bridging Example



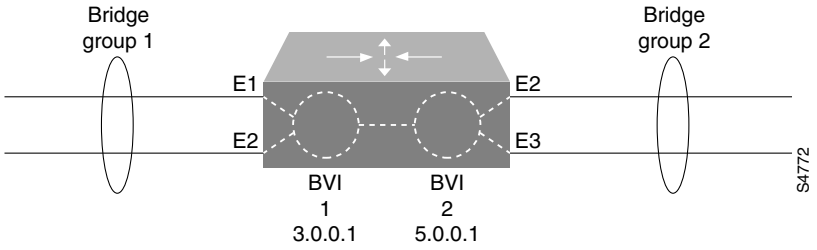
The following example shows the relevant portions of the configuration for the router:

```
appletalk routing
!
interface Ethernet 1
 ip address 5.0.0.1 255.0.0.0
 appletalk cable-range 35-35 35.1
 appletalk zone Engineering
!
interface Ethernet 2
 ip address 3.0.0.1 255.0.0.0
 bridge-group 1
!
interface Ethernet 3
 ip address 7.0.0.1 255.0.0.0
 bridge-group 1
!
interface BVI 1
 no ip address
 appletalk cable-range 33-33 33.1
 appletalk zone Accounting
!
bridge irb
bridge 1 protocol ieee
 bridge 1 route appletalk
 bridge 1 route ip
 no bridge 1 bridge ip
```

Integrated Routing and Bridging with Multiple Bridge Groups Example

In the example illustrated in Figure 15, integrated routing and bridging is used to route and bridge IP between two bridge groups.

Figure 15 Integrated Routing and Bridging with Multiple Bridge Groups



The following example shows the relevant portions of the configuration for the router in Figure 15:

```
interface Ethernet 1
 bridge-group 1
!
interface Ethernet 2
 bridge-group 1
!
interface Ethernet 3
 bridge-group 2
!
interface Ethernet 4
 bridge-group 2
!
interface BVI 1
 ip address 3.0.0.1 255.0.0.0
!
interface BVI 2
 ip address 5.0.0.1 255.0.0.0
!
bridge irb
bridge 1 protocol ieee
 bridge 1 route ip
bridge 2 protocol ieee
 bridge 2 route ip
```

Transparently Bridged VLANs Configuration Example

The following example shows the configuration for the topology in Figure 9. The “striped” VLAN is identified as security association identifier 45; the “dot” VLAN is identified as security association identifier 1008; the “sliced” VLAN is identified as security association identifier 4321. Note that the assignment of bridge group, interface, and subinterface numbers is of local significance only. You must coordinate only the configuration of a common Security Association Identifier across bridges.

Router One

```
bridge 18 protocol ieee
interface ethernet 0/1
 bridge-group 18
!
interface ethernet 0/2
 bridge-group 18
```

```
!
interface ethernet 0/3
 bridge-group 18
!
interface fddi 4/0.8
 encapsulation sde 45
 bridge-group 18
!
 bridge 54 protocol ieee

interface ethernet 1/1
 bridge-group 54
!
interface ethernet 1/2
 bridge-group 54
!
interface ethernet 1/3
 bridge-group 54
!
interface fddi 4/0.13
 encapsulation sde 1008
 bridge-group 54
!
bridge 3 protocol ieee
!
interface ethernet 2/1
 bridge-group 3
!
interface ethernet 2/2
 bridge-group 3
!
interface ethernet 2/3
 bridge-group 3
!
interface fddi 4/0.30
 encapsulation sde 4321
 bridge-group 3
```

Router Two

```
bridge 7 protocol ieee
interface ethernet 0/1
 bridge-group 7
!
interface ethernet 0/2
 bridge-group 7

interface ethernet 0/3
 bridge-group 7
!
interface ethernet 0/4
 bridge-group 7
!
interface fddi 2/0.11
 encapsulation sde 4321
 bridge-group 7


!
bridge 8 protocol ieee
interface ethernet 1/1
 bridge-group 8
!
interface ethernet 1/2
```

```
 bridge-group 8
!
interface ethernet 1/3
 bridge-group 8
!
interface ethernet 1/4
 bridge-group 8
!
interface fddi 2/0.14
 encapsulation sde 1008
 bridge-group 8
```

Router Three

```
bridge 1 protocol ieee
interface ethernet 0/1
 bridge-group 1
!
interface ethernet 0/2
 bridge-group 1
!
interface ethernet 0/3
 bridge-group 1
!
interface fddi 2/0.5
 encapsulation sde 4321
 bridge-group 1
!
bridge 6 protocol ieee
interface ethernet 1/1
 bridge-group 6
!
interface ethernet 1/2
 bridge-group 6
!
interface ethernet 1/3
 bridge-group 6
!
interface fddi 2/0.3
 encapsulation sde 45
 bridge-group 6
```

Routing Between VLANs Configuration Example

The following example shows the configuration for the topology shown in Figure 10. IP traffic is routed to and from switched VLAN domains 300, 400, and 600 to any other IP routing interface, as is IPX for VLANs 500 and 600. Because Fast Ethernet interfaces 2/1.20 and 3/1.40 are combined in bridge group 50, all other nonrouted traffic is bridged between these two subinterfaces.

```
interface FDDI 1/0.10
 ip address 131.108.1.1 255.255.255.0
 encap sde 300
!
interface Fast Ethernet 2/1.20.
 ip address 171.69.2.2 255.255.255.0
 encap isl 400
 bridge-group 50
!
interface Fast Ethernet 2/1.30
 ipx network 1000
 encap isl 500
!
interface Fast Ethernet 3/1.40
 ip address 198.92.3.3 255.255.255.0
 ipx network 1001
 encap isl 600
 bridge-group 50
!
bridge 50 protocol ieee
```

Ethernet-to-FDDI Transparent Bridging Example

The following configuration example shows the configuration commands that enable transparent bridging between Ethernet and FDDI interfaces. Transparent bridging on an FDDI interface is allowed only on the CSC-C2FCIT interface card.

```
hostname tester
!
buffers small min-free 20
buffers middle min-free 10
buffers big min-free 5
!
no ip routing
!
interface ethernet 0
 ip address 131.108.7.207 255.255.255.0
 no ip route-cache
 bridge-group 1
!
interface ethernet 2
 ip address 131.108.7.208 255.255.255.0
 no ip route-cache
 bridge-group 1
!
interface Fddi 0
 ip address 131.108.7.209 255.255.255.0
 no ip route-cache
 no keepalive
 bridge-group 1
!
bridge 1 protocol ieee
```

If the other side of the FDDI ring were an FDDI interface running in encapsulation mode rather than in transparent mode, the following additional configuration commands would be needed:

```
interface fddi 0
 fddi encapsulate
```

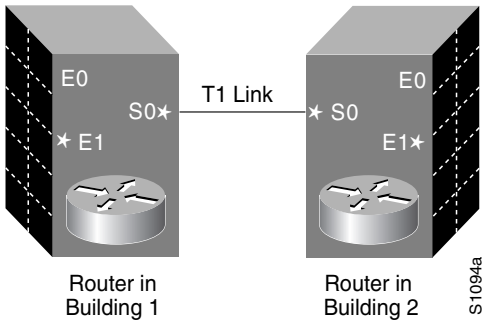
Ethernet Bridging Example

In the following example, two buildings have networks that must be connected via a T1 link. For the most part, the systems in each building use either IP or DECnet, and therefore, should be routed. There are some systems in each building that must communicate, but they can use only a proprietary protocol.

The example places two Ethernets in each building. One of the Ethernets is attached to the hosts that use a proprietary protocol, and the other is used to attach to the rest of the building network running IP and DECnet. The Ethernet attached to the hosts using a proprietary protocol is enabled for bridging to the serial line and to the other building.

Figure 16 shows an example configuration. The interfaces marked with an asterisk (*) are configured as part of spanning tree 1. The routers are configured to route IP and DECnet. This configuration permits hosts on any Ethernet to communicate with hosts on any other Ethernet using IP or DECnet. In addition, hosts on Ethernet 1 in either building can communicate using protocols not supported for routing.

Figure 16 Ethernet Bridging Configuration Example



Router/Bridge in Building 1

The configuration file for the router in Building 1 would be as follows. Note that no bridging takes place over Ethernet 0. Both IP and DECnet routing are enabled on all interfaces.

```
decnet address 3.34
interface ethernet 0
 ip address 128.88.1.6 255.255.255.0
 decnet cost 10
!
interface serial 0
 ip address 128.88.2.1 255.255.255.0
 bridge-group 1
 decnet cost 10
!
interface ethernet 1
 ip address 128.88.3.1 255.255.255.0
 bridge-group 1
 decnet cost 10
!
bridge 1 protocol dec
```

Router/Bridge in Building 2

The configuration file for the router in Building 2 is similar to Building 1:

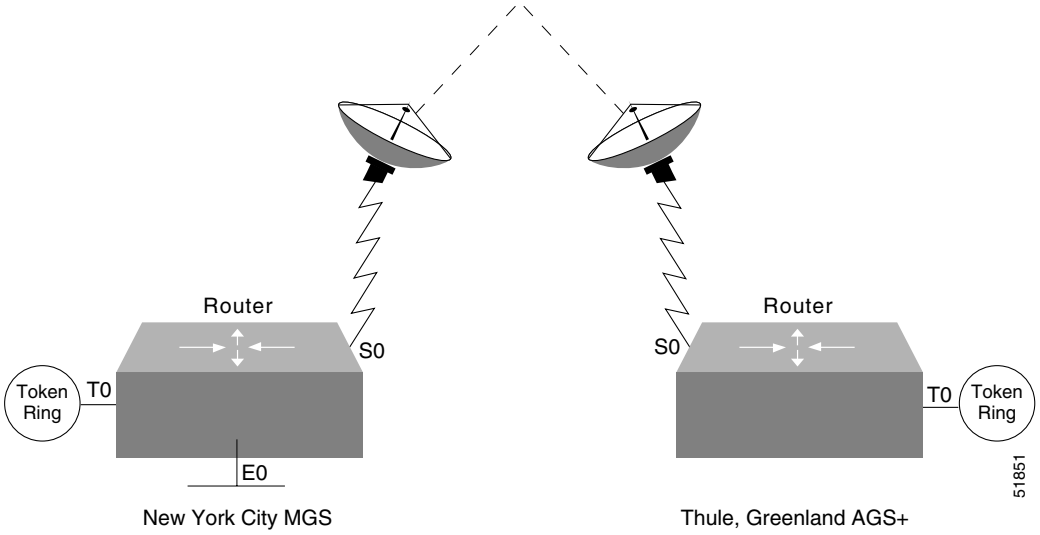
```
decnet address 3.56
!
interface ethernet 0
 ip address 128.88.11.9 255.255.255.0
 decnet cost 10
!
interface serial 0
 ip address 128.88.2.2 255.255.255.0
 bridge-group 1
 decnet cost 10
!
interface ethernet 1
 ip address 128.88.16.8 255.255.255.0
 bridge-group 1
 decnet cost 10
!
bridge 1 protocol dec
```

SRT Bridging Example

In Figure 17, a Token Ring and an Ethernet at a remote sales site in New York City must be configured to pass unroutable bridged traffic across a satellite link to the backbone Token Ring at the corporate headquarters in Thule, Greenland. IP is the only routed protocol. They are running the IEEE Spanning Tree Protocol to comply with the SRT bridging standard.

If there were source-routed traffic to bridge, the **source-bridge** command would also be used to configure source routing.

Figure 17 Network Configuration Example



Configuration for the New York City Router

```
interface tokenring 0
 ip address 150.136.1.1 255.255.255.128
 bridge-group 1
```

```
!
interface ethernet 0
 ip address 150.136.2.1 255.255.255.128
 bridge-group 1
!
interface serial 0
 ip address 150.136.3.1 255.255.255.128
 bridge-group 1
!
bridge 1 protocol ieee
```

Configuration for the Thule, Greenland Router

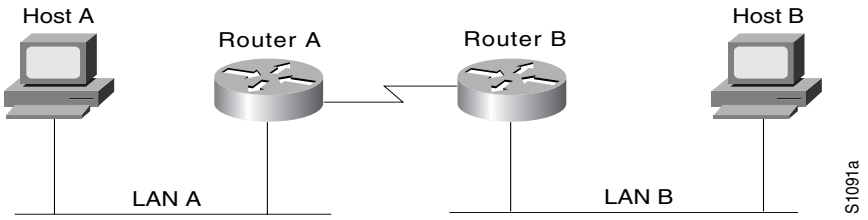
```
interface tokenring 0
 ip address 150.136.10.1 255.255.255.128
 bridge-group 1
!
interface serial 0
 ip address 150.136.11.1 255.255.255.128
 bridge-group 1
!
bridge 1 protocol ieee
```

Multicast or Broadcast Packets Bridging Example

When filtering specific MAC destination addresses, allow for multicast or broadcast packets that are required by the bridged network protocols.

Assume you are bridging IP in your network as illustrated in Figure 18.

Figure 18 Network Demonstrating Output Address List Filtering



The MAC address of Host A is 0800.0907.0207, and the MAC address of Host B is 0260.8c34.0864. The following configuration would work as expected, because input addresses work on the source address on the incoming interface:

```
access-list 700 permit 0260.8c34.0864 0000.0000.0000
access-list 700 deny 0000.0000.0000 FFFF.FFFF.FFFF
interface ethernet 0
 bridge-group 1 input-address-list 700
```

However, the following configuration might work initially but will eventually fail. The failure occurs because the configuration does not allow for an ARP broadcast with a destination address of FFFF.FFFF.FFFF, even though the destination address on the output interface is correct:

```
access-list 700 permit 0260.8c34.0864 0000.0000.0000
access-list 700 deny 0000.0000.0000 FFFF.FFFF.FFFF
interface ethernet 0
 bridge-group 1 output-address-list 700
```

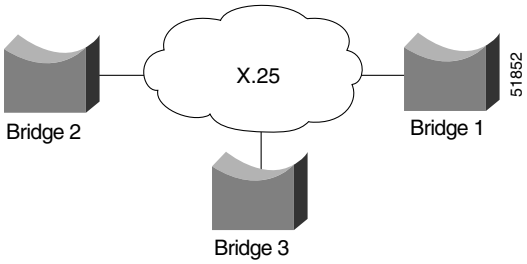
The correct access list would be as follows:

```
access-list 700 permit 0260.8c34.0864 0000.0000.0000
access-list 700 permit FFFF.FFFF.FFFF 0000.0000.0000
access-list 700 deny 0000.0000.0000 FFFF.FFFF.FFFF
interface ethernet 0
 bridge-group 1 output-address-list 700
```

X.25 Transparent Bridging Example

Figure 19 is an example configuration illustrating three bridges connected to each other through an X.25 network.

Figure 19 X.25 Bridging Examples



Following are the configuration commands for each of the bridges depicted in Figure 19:

Configuration for Bridge 1

```
interface ethernet 2
 bridge-group 5
 ip address 128.88.11.9 255.255.255.0
!
interface serial 0
 encapsulation x25
 x25 address 31370019027
 bridge-group 5
 x25 map bridge 31370019134 broadcast
 x25 map bridge 31370019565 broadcast
!
bridge 5 protocol ieee
```

Configuration for Bridge 2

```
interface serial 1
 encapsulation x25
 x25 address 31370019134
 bridge-group 5
 x25 map bridge 31370019027 broadcast
 x25 map bridge 31370019565 broadcast
!
bridge 5 protocol ieee
```

Configuration for Bridge 3

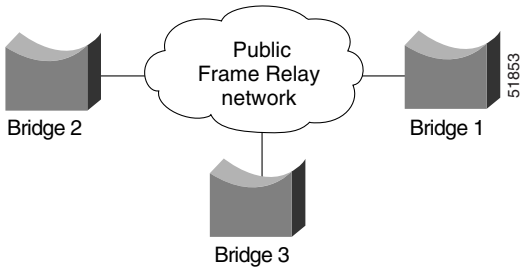
```
interface serial 0
 encapsulation x25
 x25 address 31370019565
 bridge-group 5
 x25 map bridge 31370019027 broadcast
 x25 map bridge 31370019134 broadcast
```

```
!
bridge 5 protocol ieee
```

Frame Relay Transparent Bridging Examples

Figure 20 illustrates three bridges connected to each other through a Frame Relay network.

Figure 20 Frame Relay Bridging Example



Bridging in a Frame Relay Network with No Multicasts

The Frame Relay bridging software uses the same spanning-tree algorithm as the other bridging functions, but allows packets to be encapsulated for sending across a Frame Relay network. The command specifies IP-to-DLCI address mapping and maintains a table of both the Ethernet and DLCIs. Following are the configuration commands for each of the bridges in a network that does not support a multicast facility:

Configuration for Bridge 1

```
interface ethernet 2
 bridge-group 5
 ip address 128.88.11.9 255.255.255.0
!
interface serial 0
 encapsulation frame-relay
 bridge-group 5
 frame-relay map bridge 134 broadcast
 frame-relay map bridge 565 broadcast
!
bridge 5 protocol ieee
```

Configuration for Bridge 2

```
interface serial 1
 encapsulation frame-relay
 bridge-group 5
 frame-relay map bridge 27 broadcast
 frame-relay map bridge 565 broadcast
!
bridge 5 protocol ieee
```

Configuration for Bridge 3

```
interface serial 0
 encapsulation frame-relay
 bridge-group 5
 frame-relay map bridge 27 broadcast
```

```
 frame-relay map bridge 134 broadcast
!
bridge 5 protocol ieee
```

Bridging in a Frame Relay Network with Multicasts

The multicast facility is used to learn about the other bridges on the network, eliminating the need for the **frame-relay map** commands.

Following are the configuration commands for each of the bridges in a network that supports a multicast facility:

Configuration for Bridge 1

```
interface ethernet 2
 bridge-group 5
 ip address 128.88.11.9 255.255.255.0
!
interface serial 0
 encapsulation frame-relay
 bridge-group 5
!
bridge 5 protocol ieee
```

Configuration for Bridge 2

```
interface serial 1
 encapsulation frame-relay
 bridge-group 5
!
bridge 5 protocol ieee
```

Configuration for Bridge 3

```
interface serial 0
 encapsulation frame-relay
 bridge-group 5
!
bridge 5 protocol ieee
```

Transparent Bridging over Multiprotocol LAPB Example

The following example illustrates a router configured for transparent bridging over multiprotocol LAPB encapsulation:

```
!
no ip routing
!
interface ethernet 1
 no ip address
 no mop enabled
 bridge-group 1
!
interface serial 0
 no ip address
 encapsulation lapb multi
 bridge-group 1
!
 bridge 1 protocol ieee
```

Fast-Switched Transparent Bridging over ATM Example (Cisco 7000)

The following configuration example enables fast-switched transparent bridging over ATM:

```
interface atm 4/0
 ip address 1.1.1.1 255.0.0.0
 atm pvc 1 1 1 aal5snap
 atm pvc 2 2 2 aal5snap
 atm pvc 3 3 3 aal5snap
 bridge-group 1
!
bridge 1 protocol dec
```

Transparent Bridging over DDR Examples

The following two examples differ only in the packets that cause calls to be placed. The first example specifies by protocol (any bridge packet is permitted to cause a call to be made); the second example allows a finer granularity by specifying the Ethernet type codes of bridge packets.

The first example configures the serial 1 interface for DDR bridging. Any bridge packet is permitted to cause a call to be placed.

```
no ip routing
!
interface Serial1
 no ip address
 encapsulation ppp
 dialer in-band
 dialer enable-timeout 3
 dialer map bridge name urk broadcast 8985
 dialer hold-queue 10
 dialer-group 1
 ppp authentication chap
 bridge-group 1
 pulse-time 1
!
dialer-list 1 protocol bridge permit
bridge 1 protocol ieee
bridge 1 hello 10
```

The second example also configures the serial 1 interface for DDR bridging. However, this example includes an **access-list** command that specifies the Ethernet type codes that can cause calls to be placed and a **dialer list protocol list** command that refers to the specified access list.

```
no ip routing
!
interface Serial1
 no ip address
 encapsulation ppp
 dialer in-band
 dialer enable-timeout 3
 dialer map bridge name urk broadcast 8985
 dialer hold-queue 10
 dialer-group 1
 ppp authentication chap
 bridge-group 1
 pulse-time 1
!
access-list 200 permit 0x0800 0xFFF8
!
dialer-list 1 protocol bridge list 200
bridge 1 protocol ieee
bridge 1 hello 10
```

Fast-Switched Transparent Bridging over SMDS Example

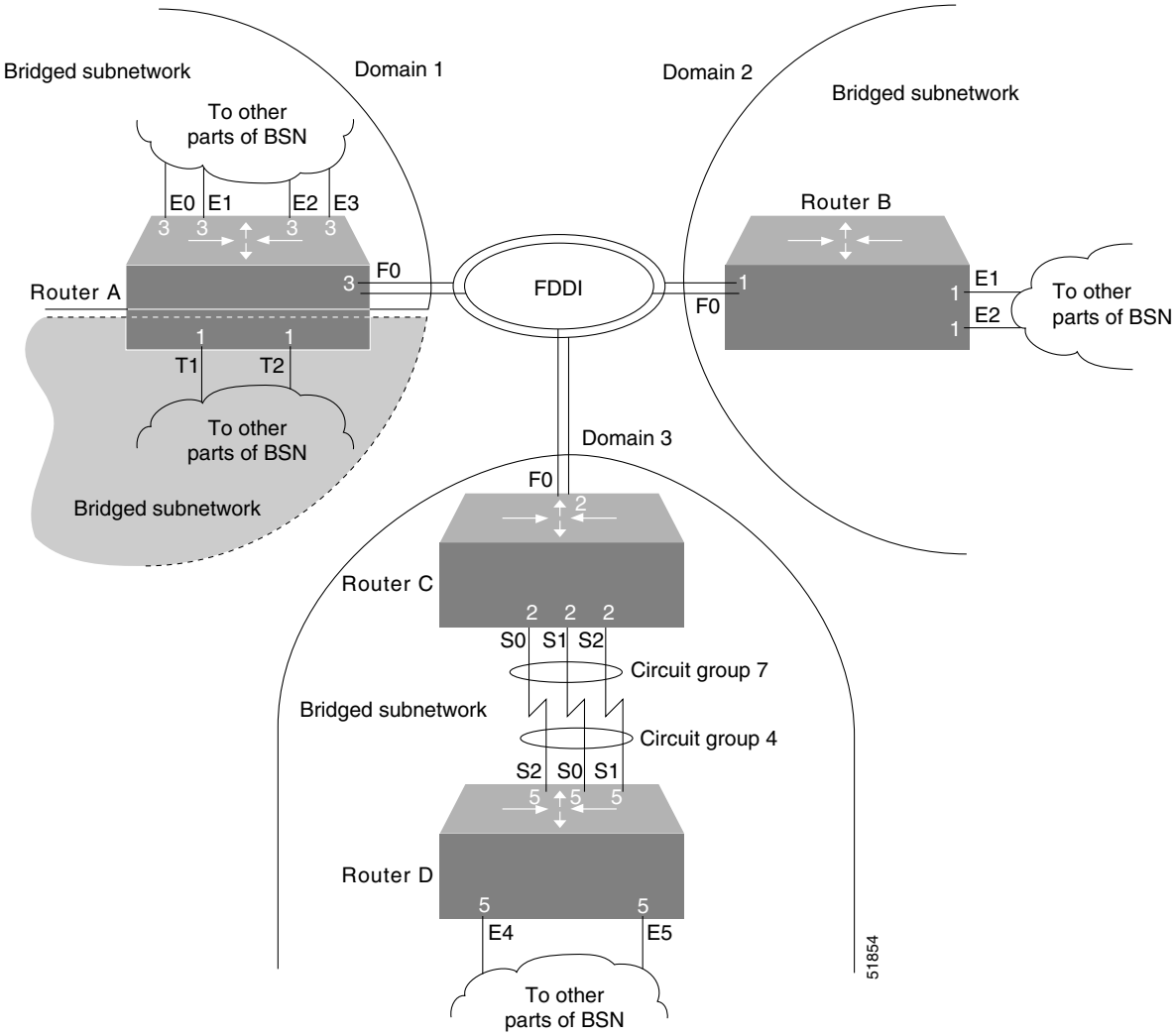
The following configuration example enables fast-switched transparent bridging over SMDS:

```
interface serial 0
 encapsulation smds
 bridge-group 1
 smds multicast bridge c141.5797.1313.ffff
```

Complex Transparent Bridging Network Topology Example

Figure 21 shows a network topology made up of four bridged subnetworks. Each bridged subnetwork is defined by the scope of a spanning tree. However, the scope of each spanning tree is not shown in detail because it is unnecessary for purposes of this discussion. Instead, it is shown by a half cloud labeled "To other parts of BSN."

Figure 21 Bridged Subnetworks with Domains



For proper bridging operation, the bridged subnetworks cannot have connections between them, but they can be connected to the same backbone. In this example, three of the four bridged subnetworks are connected to the FDDI backbone and each belongs to a separate domain.

Domains used in this topology allow the bridged subnetworks to be independent of one another while still bridging traffic onto the backbone destined for other connected bridged subnetworks. Domains can be used in this manner only if the bridged subnetworks have a single point of attachment to one another. In this case, the connection to the FDDI backbone is that single point of attachment.

Each router on which a domain is configured and that has a single point of attachment to the other bridged subnetworks, checks whether a BPDU on the backbone is its own. If the BPDU does not belong to the bridged subnetwork, the Cisco IOS software ignores the BPDU.

Separate bridged subnetworks, as in this example, allow spanning-tree reconfiguration of individual bridged subnetworks without disrupting bridging among the other bridged subnetworks.

Note To get spanning-tree information by bridge group, use the **show span** command. Included in this information is the root bridge of the spanning tree. The root bridge for each spanning tree can be any router in the spanning tree.

The routers in this network are configured for bridging and demonstrate some of the bridging features available.

Configuration for Router A

Router A demonstrates multiple bridge groups in one router for bridged traffic separation.

In Router A, the Token Ring interfaces are bridged together entirely independently of the other bridged interfaces in the router and belong to bridge group 1. Bridge group 1 does not use a bridge domain because the interfaces are bridged independently of other bridged subnetworks in the network topology and it has no connection to the FDDI backbone.

Also in Router A, the Ethernet interfaces belong to bridge group 3. Bridge group 3 has a connection to the FDDI backbone and has a domain defined for it so that it can ignore BPDUs for other bridged subnetworks.

```
interface ethernet 0
 bridge-group 3
!
interface ethernet 1
 bridge-group 3
!
interface ethernet 2
 bridge-group 3
!
interface ethernet 3
 bridge-group 3
!
interface fddi 0
 bridge-group 3
!
interface tokenring 1
 bridge-group 1
!
interface tokenring 2
 bridge-group 1
!
bridge 1 protocol ieee
bridge 3 domain 1
bridge 3 protocol ieee
```

Configuration for Router B

Router B demonstrates a simple bridge configuration. It is connected to the FDDI backbone and has domain 2 defined. As such it can bridge traffic with the other FDDI-connected BSNs. Note that bridge group 1 has no relationship to bridge group 1 in Router A; bridge groups are an organization internal to each router.

```
interface ethernet 1
 bridge-group 1
!
interface ethernet 2
 bridge-group 1
!
interface fddi 0
 bridge-group 1
!
bridge 1 domain 2
bridge 1 protocol ieee
```

Configuration for Router C

Router C and Router D combine to demonstrate load balancing by means of circuit groups. Circuit groups are used to load balance across multiple parallel serial lines between a pair of routers. The router on each end of the serial lines must have a circuit group defined. The circuit group number can be the same or can be different. In this example, they are different.

Router C and Router D are configured with the same domain, because they must understand one another's BPDUs. If they were configured with separate domains, Router D would ignore Router C's BPDUs and vice versa.

```
interface fddi 0
 bridge-group 2
!
interface serial 0
 bridge-group 2
 bridge-group 2 circuit-group 7
!
interface serial 1
 bridge-group 2
 bridge-group 2 circuit-group 7
!
interface serial 2
 bridge-group 2
 bridge-group 2 circuit-group 7
!
bridge 2 domain 3
bridge 2 protocol ieee
```

Configuration for Router D

```
interface ethernet 4
 bridge-group 5
!
interface ethernet 5
 bridge-group 5
!
interface serial 0
 bridge-group 5
 bridge-group 5 circuit-group 4
!
interface serial 1
 bridge-group 5
 bridge-group 5 circuit-group 4
!
```

```
interface serial 2
 bridge-group 5
 bridge-group 5 circuit-group 4
!
bridge 5 domain 3
bridge 5 protocol ieee
```

Fast Ethernet Subscriber Port, Frame Relay Trunk Example

The following example uses the Fast Ethernet subinterface as the subscriber port and Frame Relay as the trunk:

```
bridge 1 protocol ieee

# Form a subscriber bridge group using policy 1
#
bridge 1 subscriber-policy 1
bridge 1 protocol ieee
interface fast0.1
encapsulation isl 1
#
# Put fast0.1 into subscriber group 1
#
bridge-group 1
interface fast0.2
encapsulation isl 2
#
# put fast0.2 into subscriber group 1
#
bridge-group 1
interface serial0
encapsulation frame-relay
int s0.1 point-to-point
#
# Use PVC 155 as the signal channel for setting up connections with the access-server
#
frame-relay interface-dlci 155
#
# Set the trunk to go upstream
#
bridge-group 1 trunk
```

ATM Subscriber Ports, ATM Trunk Example

The following example uses ATM subinterfaces as the subscriber ports and the ATM as the trunk:

```
bridge 1 protocol ieee
#
# Use subscriber policy 3
#
bridge 1 subscriber-policy 3
#
# Change the ARP behavior from permit to deny
#
subscriber-policy 3 arp deny
#
# Change the multicast from permit to deny
#
subscriber-policy 3 multicast deny
```

```
int atm0
int atm0.1 point-to-point
#
# Use AAL5 SNAP encapsulation
#
atm pvc 1 0 101 aal5snap
bridge-group 1
int atm0.2
#
# Use AAL5 SNAP encapsulation
#
atm pvc 2 0 102 aal5snap
bridge-group 1

#
# Configure ATM trunk port
#
int atm1.1
#
# Use AAL5 SNAP encapsulation
#
atm pvc 1 0 101 aal5snap
#
# Specify trunk
#
bridge-group 1 trunk
```

Configuration of Transparent Bridging for PA-12E/2FE Port Adapter Example

Following is an example of a configuration for the PA-12E/2FE port adapter interface. Bridge groups 10, 20, and 30 use IEEE Spanning Tree Protocol. The first four interfaces of a PA-12E/2EF port adapter in port adapter slot 3 use bridge groups 10 and 20. Each interface is assigned to a bridge group and the shutdown state is set to up. The PA-12E/2FE port adapter supports store-and-forward or cut-through switching technology between interfaces within the same bridge group; store-and-forward is the default. In the following example, the **cut-through** command is used to configure each interface for cut-through switching of received and sent data.

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# bridge 10 protocol ieee
Router(config)# bridge 20 protocol ieee
Router(config)# bridge 30 protocol ieee

Router(config)# int fastethernet 3/0
Router(config-if)# bridge-group 10
Router(config-if)# cut-through
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Fast Ethernet3/0, changed
state to up
%LINK-3-UPDOWN: Interface Fast Ethernet3/0, changed state to up

Router(config)# int fastethernet 3/1
Router(config-if)# bridge-group 10
Router(config-if)# cut-through
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Fast Ethernet3/1, changed
state to up
%LINK-3-UPDOWN: Interface Fast Ethernet3/1, changed state to up

Router(config)# int ethernet 3/2
Router(config-if)# bridge-group 20
Router(config-if)# cut-through
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet3/2, changed state to up
%LINK-3-UPDOWN: Interface Ethernet3/2, changed state to up

Router(config)# int ethernet 3/3
Router(config-if)# bridge-group 20
Router(config-if)# cut-through
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet3/3, changed state to up
%LINK-3-UPDOWN: Interface Ethernet3/3, changed state to up
```

Configuration of IRB for PA-12E/2FE Port Adapter Example

The following example shows integrated routing and bridging enabled on the bridge groups. Bridge group 10 is assigned an IP address and subnet mask and the shutdown state is changed to up. Bridge group 10 is configured to route IP.

```
Router(config)# bridge irb
Router(config)# interface bvi 10
Router(config-if)# ip address 1.1.15.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface BVI10, changed state to up

Router(config)# bridge 10 route ip
Router(config)# exit
Router#
```

BCP Support

The Bridge Control Protocol (BCP) Support feature provides support for BCP to Cisco devices, as described in RFC 3518. The Cisco implementation of BCP is a VLAN infrastructure that does not require the use of subinterfaces to group Ethernet 802.1Q trunks and the corresponding PPP links. This approach enables users to process VLAN encapsulated packets without having to configure subinterfaces for every possible VLAN configuration.

Feature History for the BCP Support feature

Release	Modification
12.3(2)T	This feature was introduced.
12.3(4)T	This feature was modified to enhance the performance of the bridging of Ethernet packets over PPP-encapsulated interfaces. The **ppp bcp tagged-frame** command was introduced to provide the option to either enable or disable the negotiation of IEEE 802.1Q-tagged packets.

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.

Contents

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

Restrictions for BCP Support

Each individual VLAN ID can be configured only once, as either part of a single VLAN ID range or on a subinterface.

Information About BCP Support

To configure the BCP Support feature, you must understand the following concept:

- VLAN ID Ranges, page 2

VLAN ID Ranges

In the traditional, subinterface-based approach to VLANs, a subinterface is created for every necessary VLAN ID, and then the application or protocol attributes are configured on every subinterface. In the VLAN range approach, a single VLAN ID range is created, and the application or protocol attributes are configured on the range as a whole.

How to Bridge a Range of VLAN IDs

This section contains the following procedures:

- Configuring a Range of VLAN IDs, page 2
- Enabling the Negotiation of IEEE 802.1Q-Tagged Packets, page 4

Configuring a Range of VLAN IDs

In this task, you create a range of VLAN IDs and then assign the VLAN ID range to the serial interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip routing**
4. **bridge** *number* **protocol ieee**
5. **interface** *type number*
6. **vlan-range dot1q** *start-range end-range* [**native**]
7. **description** *description*
8. **bridge-group** *number*
9. **exit**
10. **interface** *type number*
11. **encapsulation ppp**
12. **bridge-group** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router(config)# enable`	Enters privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** `Router(config)# configure terminal`	Enters global configuration mode.
Step 3	**no ip routing** **Example:** `Router(config)# no ip routing`	Disables all routing.
Step 4	**bridge** *number* **protocol ieee** **Example:** `Router(config)# bridge 1 protocol ieee`	Enables bridge and spanning-tree protocols.
Step 5	**interface** *type number* **Example:** `Router(config)# interface ethernet 0`	Enters interface configuration mode. • This is the Ethernet interface that is connected to the 802.1Q trunk. Both the Ethernet interface and the serial interface must be assigned to the same bridge group.
Step 6	**vlan-range dot1q** *start-range end-range* [**native**] **Example:** `Router(config-if)# vlan-range dot1q 1 99`	Configures the range of VLAN IDs the interface is to bridge and enters VLAN range configuration mode. • Configuring the **native** keyword instructs the interface to bridge untagged (native) packets.
Step 7	**description** *description* **Example:** `Router(config-if-vlan-range)# description 1 to 99`	(Optional) Describes the VLAN ID range.
Step 8	**bridge-group** *number* **Example:** `Router(config-if-vlan-range)# bridge-group 1`	Assigns the VLAN ID range to a bridge group.
Step 9	**exit** **Example:** `Router(config-if-vlan-range)# exit`	Exits to global configuration mode.
Step 10	**interface** *type number* **Example:** `Router(config)# interface serial 1`	Enters interface configuration mode.

	Command or Action	Purpose
Step 11	**encapsulation ppp** **Example:** `Router(config-if)# encapsulation ppp`	Enables PPP on the interface.
Step 12	**bridge-group** *number* **Example:** `Router(config-if)# bridge-group 1`	Assigns the interface to a bridge group. • The serial interface must be assigned to the same bridge group as the Ethernet interface that is connected to the 802.1Q trunk.

Enabling the Negotiation of IEEE 802.1Q-Tagged Packets

In this task, you enable the negotiation of IEEE 802.1Q-tagged packets.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ppp bcp tagged-frame**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router(config)# enable`	Enters privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router(config)# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 4/0`	Enters interface configuration mode. • This is the interface that will be bridging the IEEE 802.1Q-tagged packets.
Step 4	**ppp bcp tagged-frame** **Example:** `Router(config-if)# ppp bcp tagged-frame`	Enables the negotiation of IEEE 802.1Q-tagged packets.

Configuration Examples for BCP Support

This section provides the following configuration examples:

Bridging a Range of VLAN IDs: Example

The following example bridges tagged 802.1Q packets that have VLAN IDs from 1 to 500. Ingress packets that have VLAN IDs outside of this range are dropped.

```
no ip routing
!
bridge 1 protocol ieee
!
interface ethernet 0
 vlan-range dot1q 1 500
  bridge-group 1
!
interface serial 0
 encapsulation ppp
 bridge-group 1
```

Bridging a Range of VLAN IDs over Multiple Interfaces: Example

The following example bridges two ranges of VLAN IDs. Packets with a VLAN ID from 1 to 600 are bridged by serial interface 0, and packets with a VLAN ID from 800 to 4000 are bridged by serial interface 1.

```
no ip routing
!
bridge 1 protocol ieee
bridge 2 protocol ieee
!
interface ethernet 0
 vlan-range dot1q 1 600
  bridge-group 1
 vlan-range dot1q 800 4000
  bridge-group 2
!
interface serial 0
 encapsulation ppp
 bridge-group 1
!
interface serial 1
 encapsulation ppp
 bridge-group 2
```

Bridging a Range of VLAN IDs from Untagged Packets: Example

The following example bridges untagged packets with a VLAN ID from 1 to 500:

```
interface ethernet 0
 vlan-range dot1q 1 500 native
  bridge-group 1
```

Enabling the Negotiation of IEEE 802.1Q-Tagged Packets: Example

The following example enables the negotiation of IEEE 802.1Q-tagged packets on serial interface 4/0:

```
interface serial 4/0
 ppp bcp tagged-frame
```

Additional References

The following sections provide references related to BCP support:

RFCs

RFCs	Title
3518	*Point-to-Point Protocol (PPP) Bridging Control Protocol (BCP)*
2878	*Point-to-Point Protocol (PPP) Bridging Control Protocol (BCP)*
1638	*PPP Bridging Control Protocol (BCP)*

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, and tools. 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

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Bridging Command Reference* at http://www.cisco.com/en/US/docs/ios/bridging/command/reference/br_book.html or the *Cisco IOS IBM Networking Command Reference* at http://www.cisco.com/en/US/docs/ios/ibm/command/reference/ibm_book.html. For information about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

New Commands

- **debug tbridge virtual-port**
- **ppp bcp tagged-frame**
- **vlan-id dot1q**
- **vlan-range dot1q**

Configuring Remote Source-Route Bridging

This chapter describes how to configure remote source-route bridging (RSRB). For a complete description of the RSRB commands mentioned in this chapter, refer to the "Remote Source-Route Bridging Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

In contrast to Source-Route Bridging (SRB), which involves bridging between Token Ring media only, RSRB is Cisco's first technique for connecting Token Ring networks over *non-Token Ring* network segments. (DLSw+ is Cisco's strategic method for providing this function.)

Cisco's RSRB software implementation includes the following features:

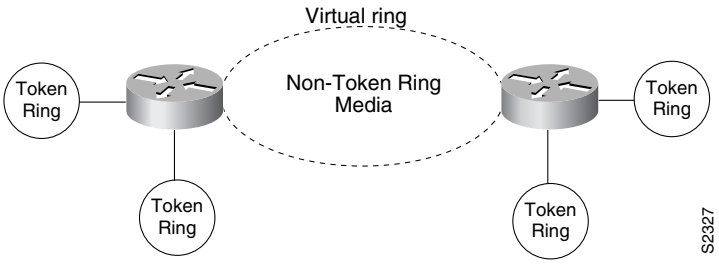
- Provides for multiple routers separated by non-Token Ring segments. Three options are available:
  - Encapsulate the Token Ring traffic inside IP datagrams passed over a Transmission Control Protocol (TCP) connection between two routers.
  - Use Fast-Sequenced Transport (FST) to transport RSRB packets to their peers without TCP or User Datagram Protocol (UDP) header or processor overhead.

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

  - Use data link layer encapsulations over a single serial line, Ethernet, Token Ring, or Fiber Distributed Data Interface (FDDI) ring connected between two routers attached to Token Ring networks.
- Provides for configurable limits to the size of the TCP backup queue.

Figure 114 shows an RSRB topology. The virtual ring can extend across any non-Token Ring media supported by RSRB, such as serial, Ethernet, FDDI, and WANs. The type of media you select determines the way you set up RSRB.

Figure 114 RSRB Topology



Note If you bridge across Token Ring media, it is recommended that you do not use RSRB. Use SRB instead. Refer to the chapter "Configuring Source-Route Bridging."

Configuration Considerations

Only use IP encapsulation over a TCP connection within complex meshed networks to support connections between peers that are separated by multiple hops and can potentially use multiple paths, and where performance is not an issue. Use direct encapsulation in point-to-point connections. In a point-to-point configuration, using TCP adds unnecessary processing overhead. Multiple peer types, however, can be combined to in a single router by following the directions for each peer type. For example, for a peer to support both TCP and FST remote-peers, you would need to define both a **source-bridge fst** peername and a **source-bridge remote-peer** command for the local router, using the same local IP address.

FST is fast-switched when it receives or sends frames from Ethernet, Token Ring, or FDDI interfaces. It is also fast-switched when it sends and receives from serial interfaces configured with the High-Level Data Link Control (HDLC) encapsulation. In all other cases, FST is slow-switched.

In cases where FST is fast-switched, in either the Cisco routers configured for FST or in the routers contained within the IP "cloud" between a pair of FST peers, only one path is used at a given time between the two FST peers. A single path greatly decreases the likelihood that frames arrive out of sequence. In the rare cases where frames do arrive out of sequence, the FST code on the receiving peer discards the out-of-order frame. Thus the Token Ring end hosts rarely lose a frame over the FST router cloud, and performance levels remain adequate.

The same conditions are true for any slow-switched topology that provides only a single path (for example, a single X.25 network cloud) between the peers. Similarly, if two slow-switched paths are of very different costs such that one always will be chosen over the other, the chances of having frames received out of sequence are also rare.

However, if two or more slow-switched paths of equal cost exist between the two routers (such as two parallel X.25 networks), the routers alternate in sending packets between the two or more equal-cost paths. This results in a high probability of frames arriving out of sequence at the receiver. In such cases, the FST code disposes of every out-of-sequence packet, leading to a large number of drops. This requires that the end hosts resend frames, greatly reducing overall throughput.

When parallel paths exist, we strongly recommend choosing one as the preferred path. Choose a preferred path by specifying a higher bandwidth for the path that contains the direct connections to the two or more parallel paths on the router.

Do not use FST when the probability exists for frames to lose their order in your network. If you have a network where frames are routinely reordered, it is better to use the TCP protocol for remote source-route bridging. TCP provides the overhead necessary to bring frames back in order on the receiving router. FST, to remain fast, does not provide for such a mechanism, and will discard out-of-order frames.

Logical Link Control, type 2 (LLC2) local acknowledgment can only be enabled with TCP remote peers (as opposed to LAN or direct serial interface remote peers) because the Cisco IOS software needs the reliability of TCP to provide the same reliability that an LLC2 LAN end-to-end connection provides. Therefore, the direct media encapsulation options for the **source-bridge remote-peer** command cannot be used.

If the LLC2 session between the local host and the router terminates on either side of the connection, the other device will be informed to terminate its connection to its local host.

If the TCP queue length of the connection between the two routers reaches 90 percent of its limit, they send Receiver-not-Ready (RNR) messages to the local hosts until the queue limit is reduced to below this limit.

The configuration of the LLC2 parameters for the local Token Ring interfaces can affect overall performance. Refer to the “Configuring LLC2 and SDLC Parameters” chapter for more details about fine-tuning your network through the LLC2 parameters.

Note Local acknowledgment for LLC2 is meant only for extreme cases in which communication is not possible otherwise. Because the router must maintain a full LLC2 session, the number of simultaneous sessions it can support before performance degrades depends on the mix of other protocols and their loads.

The routers at each end of the LLC2 session execute the full LLC2 protocol, which can result in some overhead. The decision to turn on local acknowledgment for LLC2 should be based on the speed of the backbone network in relation to the Token Ring speed. For LAN segments separated by slow-speed serial links (for example, 56 kbps), the T1 timer problem could occur more frequently. In such cases, it may be wise to turn on local acknowledgment for LLC2. For LAN segments separated by a FDDI backbone, backbone delays will be minimal; in such cases, local acknowledgment for LLC2 should not be turned on. Speed mismatch between the LAN segments and the backbone network is one criterion to be used in the decision to use local acknowledgment for LLC2.

There are some situations (such as host B failing between the time host A sends data and the time host B receives it) in which host A would behave as if, *at the LLC2 layer*, data was received when it actually was not, because the device acknowledges that it received data from host A before it confirms that host B can actually receive the data. But because both NetBIOS and SNA have error recovery in situations where an end device goes down, these higher-level protocols will resend any missing or lost data. These transaction request/confirmation protocols exist above LLC2, so they are not affected by tight timers, as is LLC2. They also are transparent to local acknowledgment.

If you are using NetBIOS applications, note that there are two NetBIOS timers—one at the link level and one at the next higher level. Local acknowledgment for LLC2 is designed to solve session timeouts at the link level only. If you are experiencing NetBIOS session timeouts, you have two options:

- Experiment with increasing your NetBIOS timers.
- Avoid using NetBIOS applications on slow serial lines.

In a configuration scenario where RSRB is configured between Router A and Router B and both routers are not routing IP, a Host connected to router A through Token Ring (or other LAN media) has no IP connectivity to router B. This restriction exists because IP datagrams received from the Host by Router A are encapsulated and sent to router B where they can only be de-encapsulated and source-bridged to a tokenring. In this scenario, IP routing is recommended. To enable the Host to reach Router B in this scenario, IP routing should be enabled on Router A's Token Ring interface to which the Host is attached.

RSRB Configuration Task List

To configure RSRB, perform the tasks in one of the following sections:

- Configuring RSRB Using Direct Encapsulation, page 4
- Configuring RSRB Using IP Encapsulation over an FST Connection, page 7
- Configuring RSRB Using IP Encapsulation over a TCP Connection, page 8
- Configuring RSRB Using TCP and LLC2 Local Acknowledgment, page 9
- Configuring Direct Frame Relay Encapsulation Between RSRB Peers, page 12
- Establishing SAP Prioritization, page 13

See the "RSRB Configuration Examples" section on page 16 for examples.

Configuring RSRB Using Direct Encapsulation

Configuring RSRB using the direct encapsulation method uses an HDLC-like encapsulation to pass frames over a single physical network connection between two routers attached to Token Rings. Use this method when you run source-route bridge traffic over point-to-point, single-hop, serial, or LAN media. Although this method does not have the flexibility of the TCP method, it provides better performance because it involves less overhead.To configure a remote source-route bridge to use a point-to-point serial line or a single Ethernet, or single FDDI hop, perform the tasks in the following sections:

- Defining a Ring Group in RSRB Context, page 4
- Identifying the Remote Peers (Direct Encapsulation), page 6
- Enabling SRB on the Appropriate Interfaces, page 6

Defining a Ring Group in RSRB Context

In our implementation of RSRB, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router with which you wish to exchange Token Ring traffic must be a member of this same ring group. These routers are referred to as remote peer bridges. The ring group is therefore made up of interfaces that reside on separate routers.

A ring group must be assigned a ring number that is unique throughout the network. It is possible to assign different interfaces on the same router to different ring groups, if, for example, you plan to administer them as interfaces in separate domains.

To define or remove a ring group, use one of the following commands in global configuration mode, as needed:

:

Command	Purpose
`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
`Router(config)# no source-bridge ring-group ring-group [virtual-mac-address]`	Removes a ring group.

Identifying the Remote Peers (Direct Encapsulation)

The interfaces that you identify as remote peer bridges must be serial, Ethernet, FDDI, or Token Ring interfaces. On a serial interface, you must use HDLC encapsulation. To identify remote-peer bridges, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge remote-peer ring-group interface interface-name [mac-address] [lf size]`	Defines the ring group and identify the interface over which to send SRB traffic to another router in the ring group.

Configure a **source-bridge remote peer** command for each peer router that is part of the virtual ring. When using direct encapsulation, you do not need to configure a local router ID.

Note If the medium being used for the direct connection is a multipoint medium (for example, Ethernet or FDDI), then you may have to specify a target Media Access Control (MAC) address for the remote peer. If so, this MAC address should be the MAC address of the interface on the remote peer that is connected to the transport medium (the medium shared by the local and remote peers).

To assign a keepalive interval to the remote source-bridging peer, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge keepalive seconds`	Defines the keepalive interval of the remote source-bridging peer.

Enabling SRB on the Appropriate Interfaces

Enable Source-Route Bridging (SRB) on each interface through which SRB traffic will pass. The value you specify in the target ring parameter should be the ring group number you have assigned to the interface. To enable SRB on an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Enables SRB on an interface.

Configuring RSRB Using IP Encapsulation over an FST Connection

Encapsulating the source-route bridged traffic inside IP datagrams passed over a FST connection between two routers is not as fast as using direct encapsulation. It does, however, outperform IP encapsulation over a TCP connection because it has lower overhead. However, this method is fast-switched and does not support any IP-level functions, including local acknowledgment or fragmentation. Nor is it suitable for use in networks that tend to reorder frame sequences.

To configure a remote source-route bridge to use IP encapsulation over an FST connection, you must perform the tasks in the following sections:

- Setting Up an FST Peer Name and Assigning an IP Address, page 7
- Identifying the Remote Peers (FST Connection), page 7
- Enabling SRB on the Appropriate Interfaces, page 8

Note FST encapsulation preserves the dynamic media-independent nature of IP routing to support SNA and NetBIOS applications.

For an example of how to configure RSRB over an FST connection, see the "RSRB Using IP Encapsulation over an FST Connection Example" section on page 19.

Setting Up an FST Peer Name and Assigning an IP Address

To set up an FST peer name and provide an IP address to be used by the local router, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge fst-peername local-interface-address`	Sets up an FST peer name and provide the local router with an IP address.

In our implementation of RSRB, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router with which you want to exchange Token Ring traffic must be a member of this same ring group. Therefore, after you set up an FST peer name, define a ring group. For more information about defining a ring group, see the "Define a Ring Group in SRB Context" section of the "Configuring Source-Route Bridging" chapter.

Identifying the Remote Peers (FST Connection)

All the routers with which you want to exchange Token Ring traffic are referred to as remote peer bridges. The remote peers can be at the other end of an FST connection. To identify the remote peers, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge remote-peer ring-group fst ip-address [lf size]`	Identifies your peers and specify an FST connection.

Specify one **source bridge remote peer** command for each peer router that is part of the virtual ring. Also specify one **source bridge remote peer** command to identify the IP address of the local router. The IP address you specify should be the IP address you want the router to reach.

You can assign a keepalive interval to the RSRB peer. Use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge keepalive seconds`	Defines the keepalive interval of the RSRB peer.

Enabling SRB on the Appropriate Interfaces

Enable SRB on each interface through which SRB traffic passes. Make the value of the target ring parameter you specify the ring group number you assigned to the interface. To enable SRB on an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Enables local SRB on a Token Ring interface.

Configuring RSRB Using IP Encapsulation over a TCP Connection

Encapsulating the source-route bridged traffic inside IP datagrams passed over a TCP connection between two routers offers lower performance, but is the appropriate method to use under the following conditions:

- You plan to connect Token Ring networks across arbitrary media including Ethernet, FDDI, serial interfaces, and X.25 networks.
- You plan to connect Token Ring networks across a multiprotocol backbone network.
- You plan to load balance over multiple, redundant paths. Using this topology, when a path fails there is no need for hosts to resend explorer packets. IP routing handles the network reconfiguration transparently to the Token Ring hosts.

To configure a remote source-route bridge to use IP encapsulation over a TCP connection, you must perform the tasks in the following sections:

- Identifying the Remote Peer (TCP Connection), page 8
- Enabling SRB on the Appropriate Interfaces, page 9

Identifying the Remote Peer (TCP Connection)

In our implementation, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router with which you want to exchange Token Ring traffic must be a member of this same ring group. For more information about defining a ring group, see the “Define a Ring Group in SRB Context” section of the “Configuring Source-Route Bridging” chapter.

To identify the remote peers, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge remote-peer ring-group tcp ip-address [lf size] [tcp-receive-window wsize] [local-ack] [priority]`	Identifies the IP address of a peer in the ring group with which to exchange source-bridge traffic using TCP.

Specify one **source-bridge remote peer** command for each peer router that is part of the virtual ring. Configure an additional **source-bridge remote peer** command to identify the IP address to be used by the local router.

You can assign a keepalive interval to the RSRB peer. Use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge keepalive seconds`	Defines the keepalive interval of the remote source-bridging peer.

Enabling SRB on the Appropriate Interfaces

To enable SRB on an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Enables local source-route bridging on a Token Ring interface.

The value of the target ring parameter you specify should be the ring group number.

Configuring RSRB Using TCP and LLC2 Local Acknowledgment

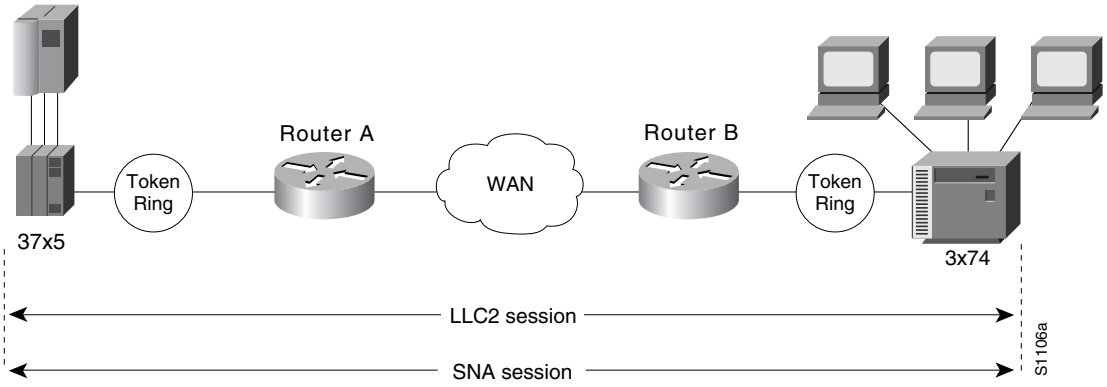
Encapsulating source-route bridged traffic inside IP datagrams that traverse a TCP connection between two routers with local acknowledgment enabled is appropriate when you have LANs separated by wide geographic distances and you want to avoid time delays, multiple resending, or loss of user sessions.

LLC2 is an ISO standard data link level protocol used in Token Ring networks. LLC2 was designed to provide reliable sending of data across LAN media and to cause minimal or at least predictable time delays. However, RSRB and WAN backbones created LANs that are separated by wide, geographic distances spanning countries and continents. As a result, LANs have time delays that are longer than LLC2 allows for bidirectional communication between hosts. Local acknowledgment addresses the problem of unpredictable time delays, multiple resending, and loss of user sessions.

In a typical LLC2 session, when one host sends a frame to another host, the sending host expects the receiving host to respond positively or negatively in a predefined period of time commonly called the *T1 time*. If the sending host does not receive an acknowledgment of the frame it sent within the T1 time, it retries a few times (normally 8 to 10 times). If there is still no response, the sending host drops the session.

Figure 115 illustrates an LLC2 session. A 37x5 on a LAN segment can communicate with a 3x74 on a different LAN segment separated via a wide-area backbone network. Frames are transported between Router A and Router B using RSRB. However, the LLC2 session between the 37x5 and the 3x74 is still end-to-end; that is, every frame generated by the 37x5 traverses the backbone network to the 3x74, and the 3x74, on receipt of the frame, acknowledges it.

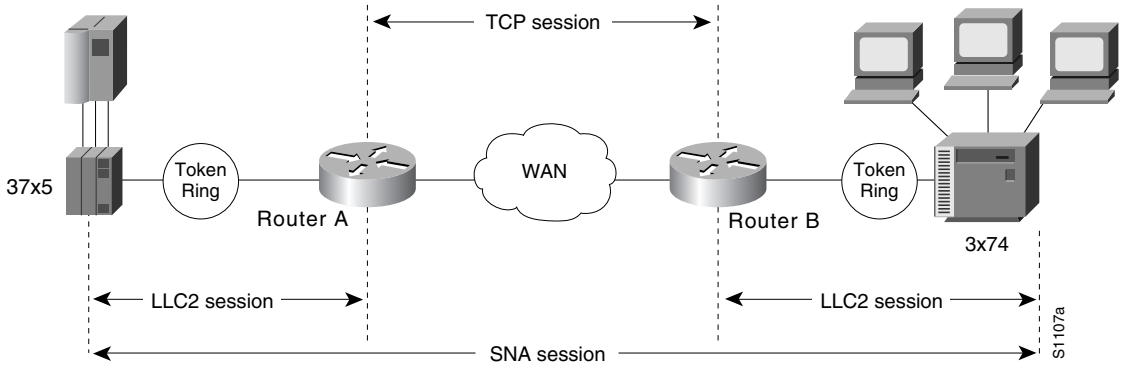
Figure 115 LLC2 Session without Local Acknowledgment



On backbone networks consisting of slow serial links, the T1 timer on end hosts could expire before the frames reach the remote hosts, causing the end host to resend. Resending results in duplicate frames reaching the remote host at the same time as the first frame reaches the remote host. Such frame duplication breaks the LLC2 protocol, resulting in the loss of sessions between the two IBM machines.

One way to solve this time delay is to increase the timeout value on the end nodes to account for the maximum transit time between the two end machines. However, in networks consisting of hundreds or even thousands of nodes, every machine would need to be reconfigured with new values. With local acknowledgment for LLC2 enabled, the LLC2 session between the two end nodes would not be end-to-end, but instead, would terminate at two local routers. Figure 116 shows the LLC2 session with the 37x5 ending at Router A and the LLC2 session with the 3x74 ending at Router B. Both Router A and Router B execute the full LLC2 protocol as part of local acknowledgment for LLC2.

Figure 116 LLC2 Session with Local Acknowledgment



With local acknowledgment for LLC2 enabled in both routers, Router A acknowledges frames received from the 37x5. The 37x5 still operates as if the acknowledgments it receives are from the 3x74. Router A looks like the 3x74 to the 37x5. Similarly, Router B acknowledges frames received from the 3x74. The

3x74 operates as if the acknowledgments it receives are from the 37x5. Router B looks like the 3x74 to 37x5. The end stations do not time out and lose sessions because the frames no longer have to travel the WAN backbone networks to be acknowledged. Instead, they are locally acknowledged by routers,

Enabling local acknowledgment for LLC2 has the following advantages:

- Local acknowledgment for LLC2 solves the T1 timer problem without having to change any configuration on the end nodes. The end nodes are unaware that the sessions are locally acknowledged. In networks consisting of hundreds or even thousands of machines, this is a definite advantage. All the frames acknowledged by the Cisco IOS software appear to the end hosts to be coming from the remote IBM machine. In fact, by looking at a trace from a protocol analyzer, one cannot say whether a frame was acknowledged by the local router or by a remote IBM machine. The MAC addresses and the RIFs generated by the Cisco IOS software are identical to those generated by the remote IBM machine. The only way to find out whether a session is locally acknowledged is to use either a **show local-ack** command or a **show source-bridge** command on the router.
- All the supervisory (RR, RNR, REJ) frames that are locally acknowledged go no farther than the router. Without local acknowledgment for LLC2, *every* frame traverses the backbone. With local acknowledgment, only data (I-frames) traverse the backbone, resulting in less traffic on the backbone network. For installations in which customers pay for the amount of traffic passing through the backbone, this could be a definite cost-saving measure. A simple protocol exists between the two *peers* to bring up or down a TCP session.

To configure a remote source-route bridge to use IP encapsulation over a TCP connection, perform the tasks in the following sections:

- Enabling LLC2 Local Acknowledgment Between Two Remote Peer Bridges, page 11
- Enabling SRB on the Appropriate Interfaces, page 11

Enabling LLC2 Local Acknowledgment Between Two Remote Peer Bridges

In our implementation, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router with which you wish to exchange Token Ring traffic must be a member of this same ring group. For more information about defining a ring group, see the “Define a Ring Group in SRB Context” section of the “Configuring Source-Route Bridging” chapter.

To enable LLC2 local acknowledgment, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge remote-peer ring-group tcp ip-address local-ack`	Enables LLC2 local acknowledgment on a per-remote-peer basis.

Use one instance of the **source-bridge remote-peer** command for each interface you configure for RSRB.

Enabling SRB on the Appropriate Interfaces

To enable SRB on an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Enables local SRB on a Token Ring interface.

The value of the target ring parameter you specify should be the ring group number.

For an example of how to configure RSRB with local acknowledgment, see the "RSRB with Local Acknowledgment Example" section on page 21.

Enabling Local Acknowledgment and Passthrough

To configure some sessions on a few rings to be locally acknowledged while the remaining sessions are passed through, use the following command in global configuration mode:

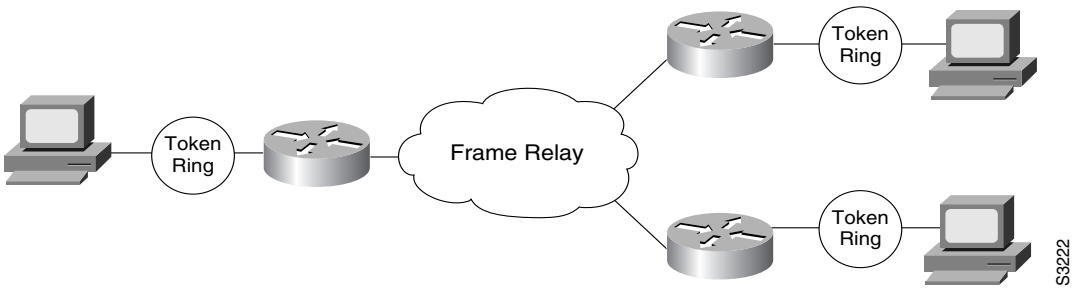
Command	Purpose
`Router(config)# source-bridge passthrough ring-group`	Configures the Cisco IOS software for passthrough.

Configuring Direct Frame Relay Encapsulation Between RSRB Peers

You can configure direct Frame Relay encapsulation to allow the RSRB peers to send RSRB protocol packets on a Frame Relay PVC. This configuration eliminates the overhead introduced by TCP/IP encapsulated Frame Relay packets.

Figure 117 illustrates direct Frame Relay encapsulation between RSRB peers.

Figure 117 RSRB Direct Frame Relay Encapsulation



The RSRB direct encapsulation design can use RFC 1490 format or Cisco Frame Relay encapsulation for routed packets.

To configure RSRB direct Frame Relay encapsulation, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# source-bridge remote-peer ring-group frame-relay interface name [mac-address] [dlci-number] [lf size]`	Specifies the serial interface on which Frame Relay is configured.
Step 2	`Router(config-if)# frame-relay map rsrb dlci-number`	Specifies the DLCI number onto which the RSRB traffic is to be mapped.

Note Direct encapsulation over Frame Relay is supported only for an encapsulation type of cisco, configured using the **encapsulation frame-relay** command.

Establishing SAP Prioritization

The SAP prioritization feature allows you to use SAP priority lists and filters to specify the priority of one protocol over another across an RSRB or SDLLC WAN.

Defining a SAP Priority List

To establish a SAP priority list, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# sap-priority-list` *number queue-keyword* [**dsap** *ds*] [**ssap** *ss*] [**dmac** *dm*] [**smac** *sm*]	Defines the priority list.
Step 2	`Router(config-if)# sap-priority` *list-number*	Defines the priority on an interface.
Step 3	`Router(config-if)# priority-group` *list-number*	Applies the priority list to an interface.

Defining SAP Filters

You can define SAP filters and NetBIOS filters on Token Ring and Ethernet interfaces.

To filter by local service access point (LSAP) address on the RSRB WAN interface, use the following global configuration commands, as needed:

Command	Purpose
`Router(config)# rsrb remote-peer` *ring-group* **tcp** *ip-address* **lsap-output-list** *access-list-number*	Filters by LSAP address (TCP encapsulation).
`Router(config)# rsrb remote-peer` *ring-group* **fst** *ip-address* **lsap-output-list** *access-list-number*	Filters by LSAP address (FST encapsulation).
`Router(config)# rsrb remote-peer` *ring-group* **interface** *name* **lsap-output-list** *access-list-number*	Filters by LSAP address (direct encapsulation).

To filter packets by NetBIOS station name on an RSRB WAN interface, use one of the following global configuration commands, as needed:

Command	Purpose
`Router(config)# rsrb remote-peer` *ring-group* **tcp** *ip-address* **netbios-output-list** *name*	Filters by NetBIOS station name (TCP encapsulation).
`Router(config)# rsrb remote-peer` *ring-group* **fst** *ip-address* **netbios-output-list** *name*	Filters by NetBIOS station name (FST encapsulation).
`Router(config)# rsrb remote-peer` *ring-group* **interface** *type* **netbios-output-list** *host*	Filters by NetBIOS station name (direct encapsulation).

RSRB Network Tuning Configuration Task List

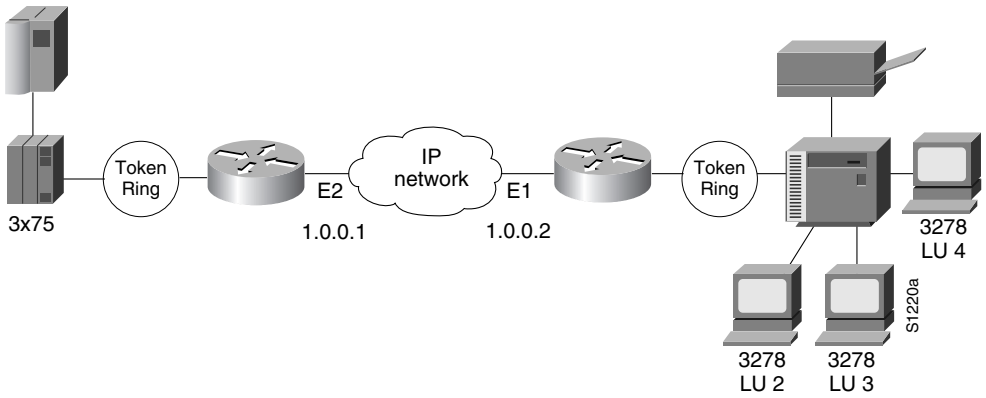
The following sections describe how to configure features that enhance network performance by reducing the number of packets that traverse the backbone network:

Prioritizing Traffic Based on SNA Local LU Addresses

You can prioritize Systems Network Architecture (SNA) traffic on an interface configured for either serial tunnel (STUN) or RSRB communication. The SNA local logical unit (LU) address prioritization feature allows SNA traffic to be prioritized according to the address of the LUs on the FID2 transmission headers. Currently, only dependent LUs are supported. The prioritization takes place on LU-LU traffic between an SNA Node type 5 or Node type 4, and Node type 2.

Figure 118 shows how SNA local address prioritization can be used.

Figure 118 SNA Local Address Prioritization



In Figure 118, the IBM mainframe is channel-attached to a 3x75 FEP, which is connected to a cluster controller via RSRB. Multiple 3270 terminals and printers, each with a unique local LU address, are then attached to the cluster controller. By applying SNA local LU address prioritization, each LU associated with a terminal or printer can be assigned a priority; that is, certain users can have terminals that have better response time than others, and printers can have lowest priority.

Note Both local acknowledgment and TCP priority features for STUN or RSRB must be turned on for SNA local address prioritization to take effect.

With the SNA local LU address prioritization feature, you can establish queueing priorities based on the address of the logical unit. To prioritize traffic, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# locaddr-priority-list list-number address-number queue-keyword [dsap ds] [dmac dm] [ssap ss] [smac sm]`	Maps LUs to TCP port numbers.
Step 2	`Router(config)# priority-list list-number protocol protocol-name queue-keyword`	Sets the queueing priority of TCP port numbers.

Enabling Class of Service

To prioritize SNA traffic across the SNA backbone network, you can enable the class of service feature. This feature is useful only between FEP-to-FEP (PU 4-to-PU 4) communication across the non-SNA backbone. It allows important FEP traffic to flow on high-priority queues.

To enable class of service, IP encapsulation over a TCP connection and LLC2 local acknowledgment must be enabled.

To enable class of service, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge cos-enable`	Enables class-of-service.

Assigning a Priority Group to an Input Interface

To assign a priority group to an input interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# locaddr-priority list-number`	Assigns a priority group to an input interface.

Configuring the Largest Frame Size

You can configure the largest frame size that is used to communicate with any peers in the ring group.

Generally, the router and the LLC2 device with which it communicates should support the same maximum SDLC I-frame size. The larger this value, the more efficiently the line is used, thus increasing performance.

Faster screen updates to 3278-style terminals often result by configuring the Token Ring FEP to send as large an I-frame as possible and then allowing the Cisco IOS software to segment the frame into multiple SDLC I-frames.

After you configure the Token Ring FEP to send the largest possible I-frame, configure the software to support the same maximum I-frame size. The default is 516 bytes and the maximum value is 8144 bytes.

To configure the largest frame size, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge largest-frame ring-group size`	Specifies the largest frame size used to communicate with any peers in the ring group.

Monitoring and Maintaining the RSRB Network

To display a variety of information about the RSRB network, use one or more of the following commands in privileged EXEC mode:

Command	Purpose
`Router# show controllers token`	Displays internal state information about the Token Ring interfaces in the system.
`Router# show interfaces`	Provides high-level statistics about the state of source bridging for a particular interface.
`Router# show local-ack`	Shows the current state of any current local acknowledgment for both LLC2 and SDLLC connections.

In addition to the EXEC-mode commands to maintain the RSRB network, you can use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge tcp-queue-max number`	Limits the size of the backup queue for RSRB to control the number of packets that can wait for transmission to a remote ring before they start being discarded.

RSRB Configuration Examples

The following sections provide RSRB configuration examples:

RSRB Direct Frame Relay Encapsulation Example

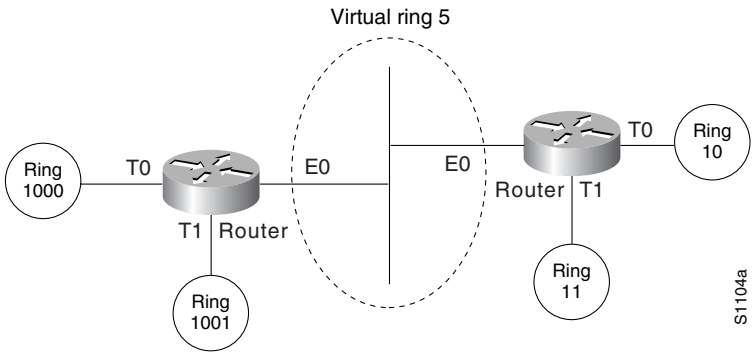
The following is the configuration file for direct Frame Relay encapsulation between RSRB peers:

```
source-bridge ring-group 200
source-bridge remote-peer 200 frame-relay interface serial0 30
!
interface serial 0
 mtu 3000
 no ip address
 encapsulation frame-relay
 clockrate 56000
 frame-relay lmi-type ansi
 frame-relay map rsrb 30
!
!
interface TokenRing 0
 ip address 10.10.10.1 255.255.255.0
 ring-speed 16
 multiring all
 source-bridge active 102 1 200
 source-bridge spanning
```

RSRB Using IP Encapsulation over a TCP Connection Example

Figure 119 illustrates two routers configured for RSRB using TCP as a transport. Each router has two Token Rings. They are connected by an Ethernet segment over which the source-route bridged traffic will pass. The first router configuration is a source-route bridge at address 131.108.2.29.

Figure 119 RSRB Using TCP as a Transport



Using TCP as the transport, the configuration for the source-route bridge at address 131.108.2.29 as depicted in Figure 119 is as follows:

```
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 131.108.2.29
source-bridge remote-peer 5 tcp 131.108.1.27
!
interface ethernet 0
 ip address 131.108.4.4 255.255.255.0
!
interface tokenring 0
 ip address 131.108.2.29 255.255.255.0
 source-bridge 1000 1 5
 source-bridge spanning
```

```
!
interface tokenring 1
 ip address 131.108.128.1 255.255.255.0
 source-bridge 1001 1 5
 source-bridge spanning
```

The configuration of the source-route bridge at 131.108.1.27 is as follows:

```
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 131.108.2.29
source-bridge remote-peer 5 tcp 131.108.1.27
!
interface ethernet 0
 ip address 131.108.4.5 255.255.255.0
!
interface tokenring 0
 ip address 131.108.1.27 255.255.255.0
 source-bridge 10 1 5
 source-bridge spanning
!
interface tokenring 1
 ip address 131.108.131.1 255.255.255.0
 source-bridge 11 1 5
 source-bridge spanning
```

RSRB/TCP Fast-Switching Configuration Example

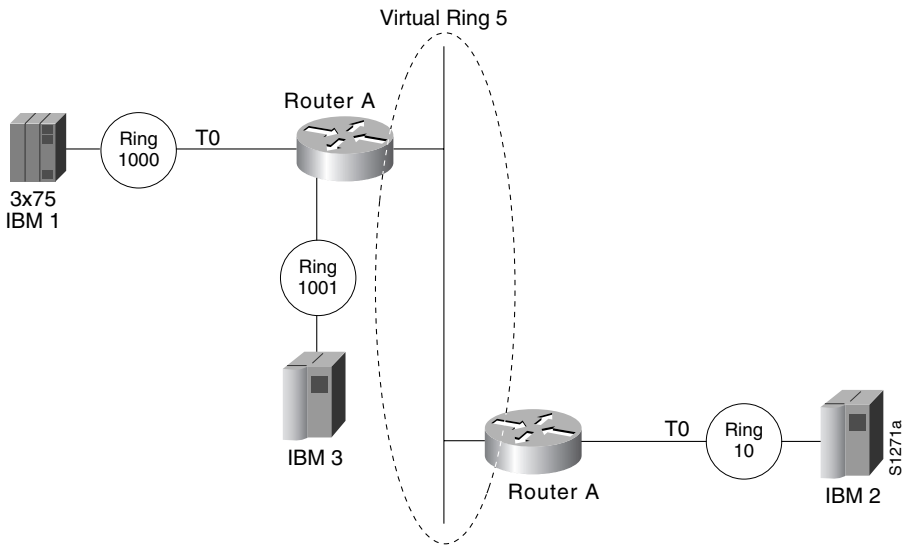
The following configuration enables RSRB/TCP fast switching:

```
source-bridge ring group 100
```

RSRB Using IP Encapsulation over an FST Connection Example

Figure 120 shows two routers connecting IBM hosts on Token Rings through an Ethernet backbone.

Figure 120 RSRB Using FST as a Transport



This configuration example enables IP encapsulation over an FST connection. In this configuration, the **source-bridge fst-peername** global configuration command is used to provide an IP address for the local router. The **source-bridge ring-group** global configuration command is used to define a ring group. The **source-bridge remote peer** command with the **fst** option is used to associate the remote peer's IP address with the router's ring group and specify the remote peer's remote source-route bridging protocol version number. Because all FST peers support version 2 RSRB, the **version** keyword is always specified.

The configuration of the source-route bridge at 131.108.2.29 is as follows:

```
source-bridge fst-peername 131.108.2.29
source-bridge ring-group 5
source-bridge remote-peer 5 fst 131.108.1.27
!
interface ethernet 0
 ip address 131.108.4.4 255.255.255.0
!
interface tokenring 0
 ip address 131.108.2.29 255.255.255.0
 source-bridge 1000 1 5
 source-bridge spanning
!
interface tokenring 1
 ip address 131.108.128.1 255.255.255.0
 source-bridge 1001 1 5
 source-bridge spanning
```

The configuration of the source-route bridge at 131.108.1.27 is as follows:

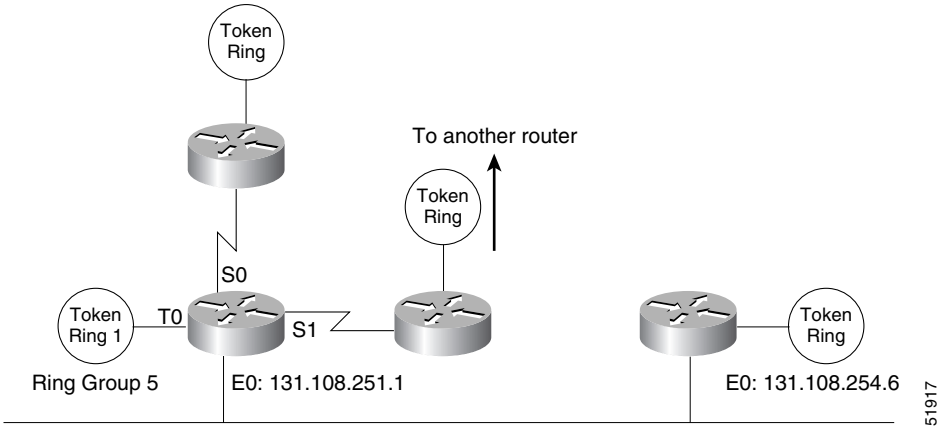
```
source-bridge fst-peername 131.108.1.27
source-bridge ring-group 5
source-bridge remote-peer 5 fst 131.108.2.29
!
interface ethernet 0
```

```
 ip address 131.108.4.5 255.255.255.0
!
interface tokenring 0
 ip address 131.108.1.27 255.255.255.0
 source-bridge 10 1 5
 source-bridge spanning
!
interface tokenring 1
 ip address 131.108.131.1 255.255.255.0
 source-bridge 11 1 5
 source-bridge spanning
```

RSRB Using All Types of Transport Methods Example

Figure 121 shows a router configured for RSRB using all types of transport methods.

Figure 121 RSRB Using All Types of Transport Methods



The configuration for the network in Figure 121 is as follows:

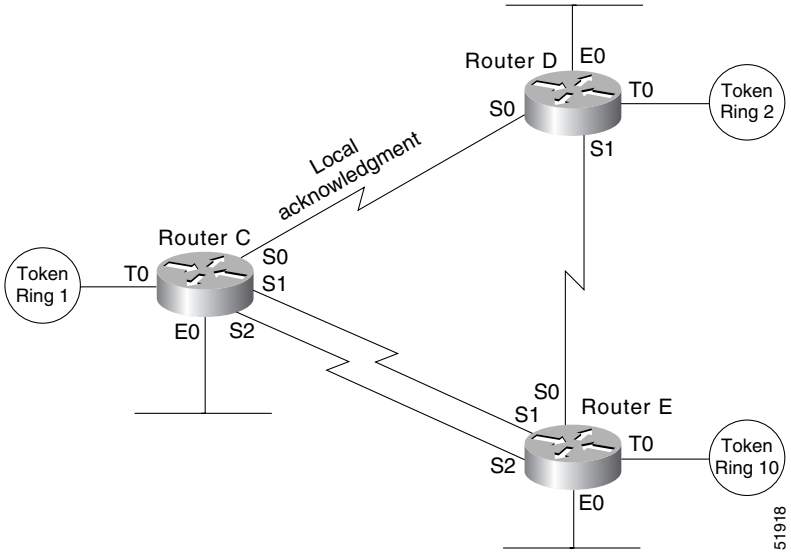
```
source-bridge fst-peername 131.108.251.1
source-bridge ring-group 5
source-bridge remote-peer 5 interface serial0
source-bridge remote-peer 5 interface serial1
source-bridge remote-peer 5 interface Ethernet0 0000.0c00.1234
source-bridge remote-peer 5 tcp 131.108.251.1
source-bridge remote-peer 5 fst 131.108.252.4
source-bridge remote-peer 5 tcp 131.108.253.5
!
interface tokenring 0
 source-bridge 1 1 5
 source-bridge spanning
!
interface ethernet 0
 ip address 131.108.251.1 255.255.255.0
```

The two peers using the serial transport method only function correctly if routers at the other end of the serial line have been configured to use the serial transport. The peers must also belong to the same ring group.

RSRB with Local Acknowledgment Example

In Figure 122, a triangular configuration provides the maximum reliability with minimal cost, and one of the links is doubled to gain better bandwidth. In addition to IP and SRB traffic, AppleTalk is also routed between all the sites. In this configuration, all the sessions between Router C and Router D are locally acknowledged. All the sessions between Router C and Router E are not locally acknowledged and are configured for normal remote source-route bridging. This example shows that not every peer must be locally acknowledged and that local acknowledgment can be turned on or off at the customer's discretion.

Figure 122 RSRB with Local Acknowledgment—Simple Configuration



The configuration for each of the routers in Figure 122 follows.

Router C

```
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1
source-bridge remote-peer 5 tcp 132.21.2.6 local-ack
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface tokenring 0
 ip address 132.21.1.1 255.255.255.0
 source-bridge 1 1 5
 source-bridge spanning
 multiring all


!
interface ethernet 0
 ip address 132.21.4.25 255.255.255.0
 appletalk address 4.25
 appletalk zone Twilight
!
interface serial 0
 ip address 132.21.16.1 255.255.255.0
 appletalk address 16.1
```

```
 appletalk zone Twilight
!
interface serial 1
 ip address 132.21.17.1 255.255.255.0
 appletalk address 17.1
 appletalk zone Twilight
!
interface serial 2
 ip address 132.21.18.1 255.255.255.0
 appletalk address 18.1
 appletalk zone Twilight
!
 router igrp 109
 network 132.21.0.0
!
 hostname RouterC
```

Router D

```
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1 local-ack
source-bridge remote-peer 5 tcp 132.21.2.6
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface tokenring 0
 ip address 132.21.2.6 255.255.255.0
 source-bridge 2 1 5
 source-bridge spanning
 multiring all
!
interface ethernet 0
 ip address 132.21.5.1 255.255.255.0
 appletalk address 5.1
 appletalk zone Twilight
!
interface serial 0
 ip address 132.21.16.2 255.255.255.0
 appletalk address 16.2
 appletalk zone Twilight
!
interface serial 1
 ip address 132.21.19.1 255.255.255.0
 appletalk address 19.1
 appletalk zone Twilight
!
router igrp 109
 network 132.21.0.0
!
hostname RouterD
```

Router E

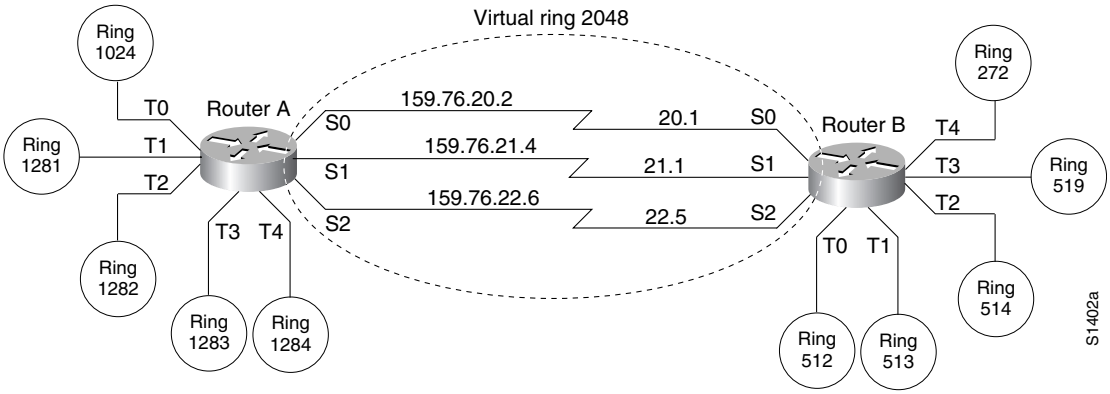
```
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1
source-bridge remote-peer 5 tcp 132.21.2.6
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface tokenring 0
 ip address 132.21.10.200 255.255.255.0
 source-bridge 10 1 5
```

```
 source-bridge spanning
 multiring all
!
interface ethernet 0
 ip address 132.21.7.1 255.255.255.0
 appletalk address 7.1
 appletalk zone Twilight
!
interface serial 0
 ip address 132.21.19.2 255.255.255.0
 appletalk address 19.2
 appletalk zone Twilight
!
interface serial 1
 ip address 132.21.17.2 255.255.255.0
 appletalk address 17.2
 appletalk zone Twilight
!
interface serial 2
 ip address 132.21.18.2 255.255.255.0
 appletalk address 18.2
 appletalk zone Twilight
!
 router igrp 109
 network 132.21.0.0
!
hostname RouterE
```

RSRB with Local Acknowledgment and Passthrough Example

Figure 123 shows two routers configured for RSRB with local acknowledgment and passthrough over the three serial lines that connect these routers. In the example, five Token Rings connect to each of these routers.

Figure 123 Network Topology for RSRB with Local Acknowledgment and Passthrough



The configuration files for each of these routers follows.

Router A

```
source-bridge ring-group 2048
source-bridge remote-peer 2048 tcp 159.76.1.250 local-ack version 2
source-bridge remote-peer 2048 tcp 159.76.7.250 version 2
source-bridge passthrough 1281
source-bridge passthrough 1282
source-bridge passthrough 1283
source-bridge passthrough 1284
!
interface tokenring 0
 ip address 159.76.7.250 255.255.255.0
 llc2 ack-max 1
 llc2 t1-time 1800
 llc2 idle-time 29000
 llc2 ack-delay-time 5
 source-bridge 1024 1 2048
 source-bridge spanning
 early-token-release
 multiring all
!
interface tokenring 1
 ip address 159.76.8.250 255.255.255.0
 clns-speed 4
 clns mtu 464
 source-bridge 1281 1 2048
 source-bridge spanning
 multiring all


!
interface tokenring 2
 ip address 159.76.9.250 255.255.255.0
 ring-speed 4
 clns mtu 4464
 source-bridge 1282 1 2048
```

```
 source-bridge spanning
 multiring all
!
interface tokenring 3
 ip address 159.76.10.250 255.255.255.0
 ring speed 4
 clns mtu 4464
 source-bridge 1283 1 2048
 source-bridge spanning
 multiring all
!
interface tokenring 4
 ip address 159.78.11.250 255.255.255.0
 ring speed 4
 clns mtu 4464
 source-bridge 1284 1 2048
 source-bridge spanning
 multiring all
!
interface serial 0
 ip address 159.76.20.2 255.255.255.0
!
interface serial 1
 ip address 159.76.21.4 255.255.255.0
!
interface serial 2
 ip address 159.76.22.6 255.255.255.0
 shutdown
! interface serial 3
 no ip address
 shutdown
```

Router B

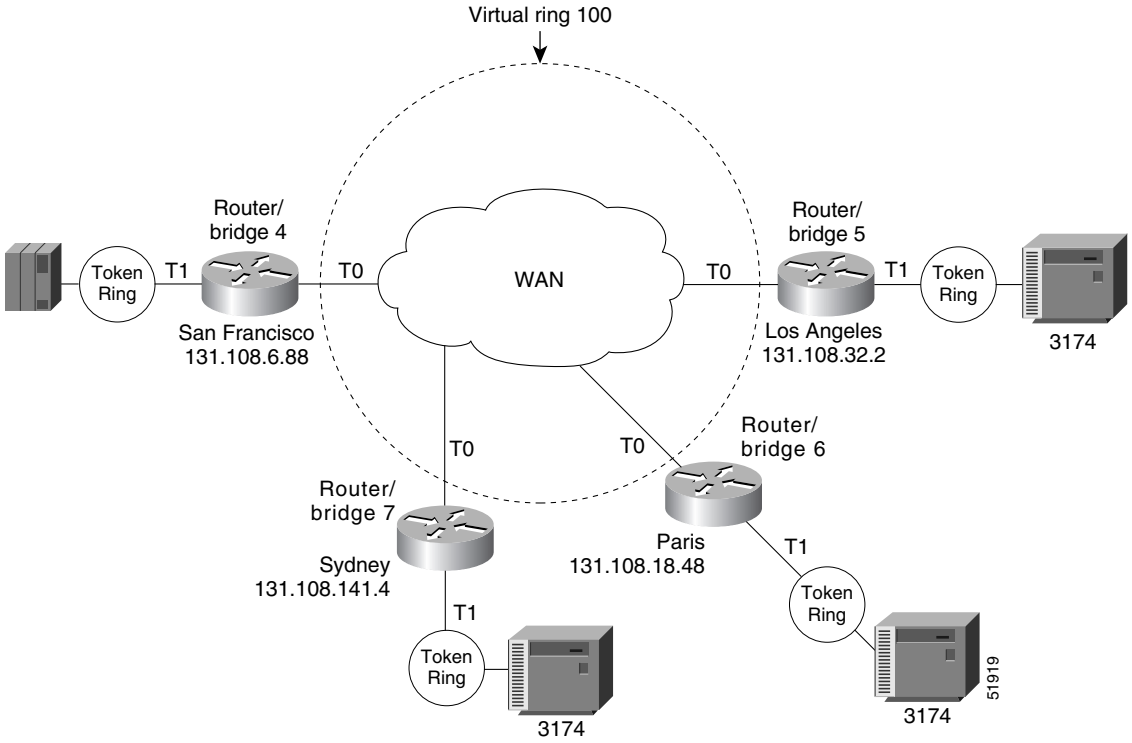
```
source-bridge ring-group 2048
source-bridge remote-peer 2048 tcp 159.76.1.250 version 2
source-bridge remote-peer 2048 tcp 159.76.7.250 local-ack version 2
!
interface tokenring 0
 ip address 159.76.1.250 255.255.255.0
 llc2 ack-max 2
 llc2 t1-time 1900
 llc2 idle-time 29000
 llc2 ack-delay-time 5
 source-bridge 512 1 2048
 source-bridge spanning
 early-token-release
 multiring all
!
interface tokenring 1
 ip address 159.76.2.250 255.255.255.0
 ring-speed 16
 clns mtu 8136


!
 source-bridge 513 1 2048
 source-bridge spanning
 early-token-release
 multiring all
!
interface tokenring 2
 ip address 159.76.3.250 255.255.255.0
 ring speed 16
```

```
 clns mtu 8136
 source-bridge 514 1 2048
 source-bridge spanning
 early-token-release
 multiring all
!
interface tokenring 3
 ip address 159.76.4.250 255.255.255.0
 ring-speed 4
 clns mtu 4464
 source-bridge 519 2 2043
 source-bridge spanning
 multiring all
!
interface tokenring 4
 ip address 159.76.5.250 255.255.255.0
 ring-speed 4
 clns mtu 4464
 source-bridge 272 2 2048
 source-bridge spanning
 multiring all
!
interface serial 0
 ip address 159.76.20.1 255.255.255.0
!
interface serial 1
 ip address 159.76.21.3 255.255.255.0
!
interface serial 2
 ip address 159.76.22.5 255.255.255.0
!
interface serial 3
 no ip address
 shutdown
```

Local Acknowledgment for LLC2 Example

Figure 124 shows an IBM FEP located in San Francisco communicating with IBM 3174 cluster controller hosts in Sydney, Paris, and Los Angeles. The session between the FEP and the IBM 3174 system in Los Angeles is not locally terminated, because the distance is great enough to cause timeouts on the line. However, the sessions to Paris and Sydney are locally terminated.

Figure 124 RSRB with Local Acknowledgment—Complex Configuration



The configuration for each of these routers follows.

Router/Bridge 4 in San Francisco

```
source-bridge ring-group 100
! use direct encapsulation across serial link to Los Angeles
 source-bridge remote-peer 100 direct 131.108.32.2
 ! use fast sequenced transport with local termination to Paris
 source-bridge remote-peer 100 tcp 131.108.18.48 local-ack
 ! use tcp encapsulation with local termination to Sydney
 source-bridge remote-peer 100 tcp 131.108.141.4 local-ack
 !
 interface tokenring 0
 ! source ring 1, bridge 4, destination ring 100
 source-bridge 1 4 100
 ! receive up to seven frames before sending an acknowledgment
 llc2 ack-max 7
 ! allow a 30 msec delay before I-frames must be acknowledged
 llc2 ack-delay-time 30
 !
 interface tokenring 1
 ! source ring 100, bridge 4, destination ring 1
 source-bridge 100 4 1
```

Router/Bridge 7 in Sydney

```
source-bridge ring-group 100
 ! use tcp encapsulation with local termination from Sydney
 source-bridge remote-peer 100 tcp 131.108.6.88 local-ack
 interface tokenring 0
 ! source ring 1, bridge 7, destination ring 100
 source-bridge 1 7 100
 ! receive up to seven frames before sending an acknowledgment
```

```
 llc2 ack-max 7
 ! allow a 30 msec delay before I-frames must be acknowledged
 llc2 ack-delay-time 30
 !
 interface tokenring 1
 ! source ring 100, bridge 7, destination ring 1
 source-bridge 100 7 1
```

Router/Bridge 6 in Paris

```
source-bridge ring-group 100
 ! use fast sequenced transport with local termination from Paris
 source-bridge remote-peer 100 tcp 131.108.6.88 local-ack
 interface tokenring 0
 ! source ring 1, bridge 6, destination ring 100
 source-bridge 1 6 100
 ! receive up to seven frames before sending an acknowledgment
 llc2 ack-max 7
 ! allow a 30 msec delay before I-frames must be acknowledged
 llc2 ack-delay-time 30
 !
 interface tokenring 1
 ! source ring 100, bridge 6, destination ring 1
 source-bridge 100 6 1
```

Router/Bridge 5 in Los Angeles

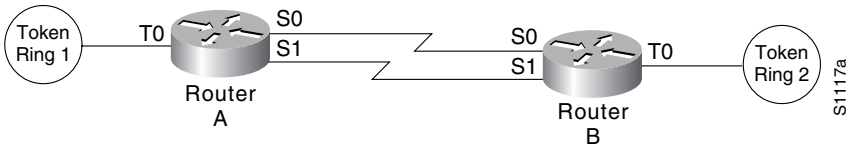
```
source-bridge ring-group 100
 ! use direct encapsulation across serial link from Los Angeles
 source-bridge remote-peer 100 direct 131.108.6.88

 interface tokenring 0
 ! source ring 1, bridge 5, destination ring 100
 source-bridge 1 5 100
 ! receive up to seven frames before sending an acknowledgment
 llc2 ack-max 7
 ! allow a 30 msec delay before I-frames must be acknowledged
 llc2 ack-delay-time 30
 !
 interface tokenring 1
 ! source ring 100, bridge 5, destination ring 1
 source-bridge 100 5 1
```

Note Both peers need to be configured for LLC2 local acknowledgment. If only one is so configured, undesirable results occur.

IP for Load Sharing over RSRB Example

As Figure 125 shows, two routers are connected by two serial lines. Each is configured as a basic remote dual-port bridge, but extended to include both reliability and IP load sharing. When both serial lines are up, traffic is split between them, effectively combining the bandwidth of the connections. If either serial line goes down, all traffic is routed to the remaining line with no disruption. This happens transparently with respect to the end connections, unlike other source-route bridges that would abort those connections.

Figure 125 RSRB—Simple Reliability



The sample configuration files that enable this configuration follow.

Configuration for Router/Bridge A

```
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 204.31.7.1
source-bridge remote-peer 5 tcp 204.31.8.1
!
interface tokenring 0
 ip address 204.31.7.1 255.255.255.0
 source-bridge 1 1 5
 source-bridge spanning
 multiring all
!
interface serial 0
 ip address 204.31.9.1 255.255.255.0
!
interface serial 1
 ip address 204.31.10.1 255.255.255.0
!
 router igrp 109
 network 204.31.7.0
 network 204.31.9.0
 network 204.31.10.0
!
hostname RouterA
```

Configuration for Router/Bridge B

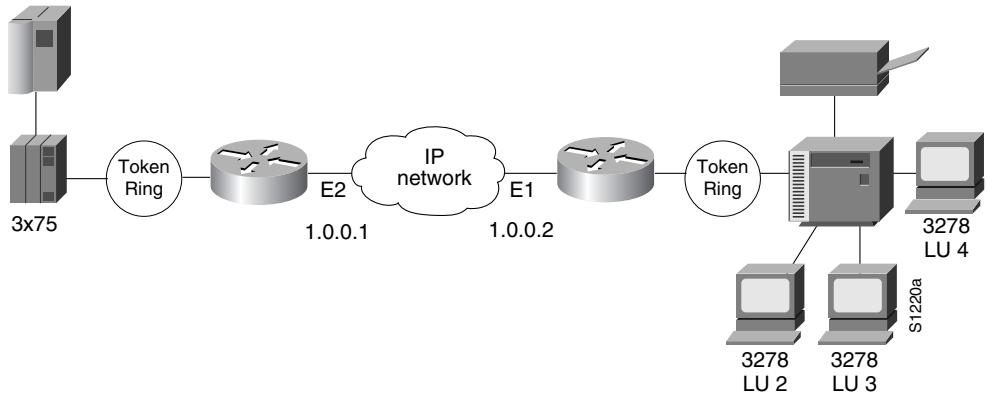
```
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 204.31.7.1
source-bridge remote-peer 5 tcp 204.31.8.1
!
interface tokenring 0
 ip address 204.31.8.1 255.255.255.0
 source-bridge 2 1 5
 source-bridge spanning
 multiring all


!
interface serial 0
 ip address 204.31.9.2 255.255.255.0
!
```

```
interface serial 1
 ip address 204.31.10.2 255.255.255.0
!
 router igrp 109
 network 204.31.8.0
 network 204.31.9.0
 network 204.31.10.0
!
 hostname RouterB
```

Configuring Priority for Locally Terminated Token Ring Interfaces in RSRB Example

Figure 126 shows a network that uses RSRB to bridge Token Ring traffic.

Figure 126 RSRB Configuration Example



The configuration for each of the routers in Figure 126 follows.

Router/Bridge A

```
source-bridge ring-group 2624
source-bridge remote-peer 2624 tcp 1.0.0.1
source-bridge remote-peer 2624 tcp 1.0.0.2 local-ack priority
!
interface tokenring 0
 source-bridge 2576 8 2624
 source-bridge spanning
 multiring all
 locaddr-priority 1
!
interface ethernet 0
 ip address 1.0.0.1 255.255.255.0
 priority-group 1


!
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
priority-list protocol ip high tcp 1996
priority-list protocol ip medium tcp 1987
```

```
priority-list protocol ip normal tcp 1988
priority-list protocol ip low tcp 1989
```

Router/Bridge B

```
source-bridge ring-group 2624
source-bridge remote-peer 2624 tcp 1.0.0.2
source-bridge remote-peer 2624 tcp 1.0.0.1 local-ack priority
!
interface tokenring 0
 source-bridge 2626 8 2624
 source-bridge spanning
 multiring all
 locaddr-priority 1
!
interface ethernet 0
 ip address 1.0.0.2 255.255.255.0
 priority-group 1
!
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
priority-list protocol ip high tcp 1996
priority-list protocol ip medium tcp 1987
priority-list protocol ip normal tcp 1988
priority-list protocol ip low tcp 1989
```

Configuring Token Ring Inter-Switch Link

This chapter explains how to configure Token Ring Inter-Switch Link (TRISL) on Cisco routers. The chapter describes TRISL in the context of the Inter-Switch Link (ISL) protocol and the Token Ring VLAN concept.

For a complete description of the Token Ring Inter-Switch Link commands in this chapter, refer to the "Token Ring Inter-Switch Link Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online. For information on how Token Ring VLANs are implemented on switches, refer to the *Catalyst Token Ring Switching Implementation Guide,* the *Catalyst 5000 Series Token Ring Configuration Notes,* the *Catalyst 3900 Token Ring Switching User Guide,* and the *Catalyst 3920 Token Ring Switching User Guide*.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

Cisco's TRISL Implementation

This section contains information related to Cisco's implementation of TRISL that you should understand before you proceed to the "TRISL Configuration Task List" section on page 5.

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

ISL and TRISL

ISL is a Layer 2 protocol that enables switches and routers to transport Ethernet frames from multiple VLANs across Fast Ethernet or Gigabit Ethernet links. Cisco's TRISL protocol extends the ISL model to include the transport of Token Ring frames from multiple VLANs across these same links.

TRISL support on Cisco routers provides inter-VLAN routing and bridging across a 100-Mb Fast Ethernet link. ISL and TRISL together provide routing and bridging between Token Ring and Ethernet LANs, ELANS, and VLANs.

TRISL is supported on the following platforms with any one of the following port adapters:

- Cisco 7500 or Cisco 7200 series routers
  - Two-port Fast Ethernet/ISL 100BaseTX
  - Two-port Fast Ethernet/ISL 100BaseFX
  - One-port Fast Ethernet 100BaseTX
  - One-port Fast Ethernet 100BaseFX
- Cisco 4500 or 4700 series routers
  - NM-1FE
- Cisco 3600 or 2600 series routers
  - NM-1FE1CE1
  - NM-1FE1CT1
  - NM-1FE1R2W
  - NM-1FE2CE1
  - NM-1FE2CT1
  - NM-1FE2W
  - NM-2FE2W

Note The two-port Fast Ethernet/ISL port adapters support frame sizes up to 17800 bytes and the one-port Fast Ethernet port adapters support a frame size of up to 1500 bytes.

TRISL provides the following capabilities and features, which will be described in the "TRISL Configuration Task List" section on page 5 and the "TRISL Configuration Examples" section on page 11:

- IP routing for source-routed and non-source-routed frames between TRISL VLANs and any LAN, ELAN, or VLAN.
- IPX routing for source-routed and non-source-routed frames between TRISL VLANs and any LANs, ELANs, or VLANs.
- Source-Route Bridging (SRB) between TRISL VLANs and SRB-capable LANs, ELANs, or VLANs.
- Source-Route Transparent Bridging (SRT) between TRISL VLANs and SRT-capable LANs, ELANs, or VLANs.
- Source-Route Translational Bridging (SR/TLB) between TRISL VLANs and Ethernet LANs, ELANs, or VLANs.

- Duplicate Ring Protocol (DRiP), which prevents external loops that could result if the router's virtual ring number were duplicated elsewhere in the network.

Note VLAN Trunk Protocol (VTP) is currently not supported for TRISL on the routers.

Token Ring VLANs

A VLAN is essentially a broadcast domain. In transparent bridging, there is only one type of broadcast frame and, therefore, only one level of broadcast domain and one level of VLAN. In source routing, however, there are two types of broadcast frames:

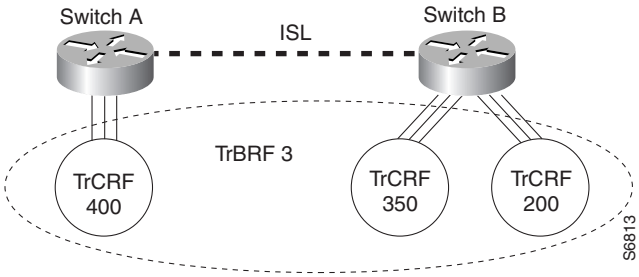
- Those that are confined to a single ring
- Those that traverse the bridged domain

Therefore, there are two levels of VLANs in a Token Ring switched network.

The first level is the Token Ring Concentrator Relay Function (TrCRF). At this level, the VLAN is a logical ring and, as such, is assigned a ring number. On a Token Ring switch, the logical ring (TrCRF) contains one or more physical ports. On a router, the logical ring (TrCRF) does not contain any physical ports, but rather is used only in processing source-routed traffic to terminate the routing information field (RIF).

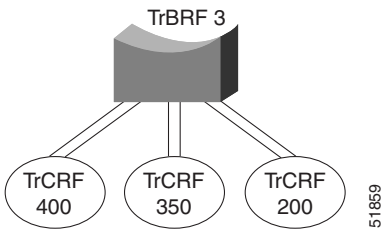
The second level is the Token Ring Bridge Relay Function (TrBRF). This is the parent VLAN to which TrCRF VLANs are assigned. At this level, the VLAN is a logical bridge and, as such, is assigned a bridge number. The logical bridge (TrBRF) contains the virtual ports that establish a connection between the TrBRF and its TrCRFs. The TrBRF can be extended across a network of switches and routers via ISL, as shown in Figure 52.

Figure 52 Physical View of Switches Interconnected via ISL



When you extend the TrBRF across an ISL link, you are essentially extending the bridge across devices, as shown in Figure 53.

Figure 53 Logical View of Switches Interconnected via ISL

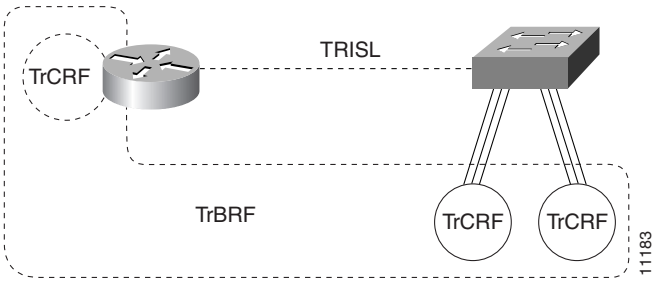


Therefore, if you use source-route bridging between the TrCRFs that belong to the TrBRF, only one hop appears in the RIF.

Traffic is switched between the ports in a TrCRF and bridged via SRB or SRT between the TrCRFs in a TrBRF.

Figure 54 illustrates a TrBRF that contains TrCRFs on both a router and a switch.

Figure 54 TrCRFs in a TrBRF



TRISL Configuration Task List

To configure and monitor TRISL in your network, perform one or more of the following tasks:

- Configuring IP Routing over TRISL, page 5
- Configuring Hot Standby Router Protocol over TRISL, page 6
- Configuring IPX Routing over TRISL, page 7
- Configuring Source-Route Bridging over TRISL, page 8
- Configuring Source-Route Transparent Bridging over TRISL, page 8
- Configuring Source-Route Translational Bridging over TRISL, page 9

See the "TRISL Configuration Examples" section on page 11 for examples.

Configuring IP Routing over TRISL

The IP routing over TRISL VLANs feature extends IP routing capabilities to include support for routing IP frame types in VLAN configurations. To configure IP routing over TRISL, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip routing`	Enables IP routing on the router.
Step 2	`Router(config)# interface` *type slot/port.subinterface-number*	Specifies the subinterface on which TRISL will be used.
Step 3	`Router(config-if)# encapsulation tr-isl trbrf-vlan` *vlanid* `bridge-num` *bridge-number*	Defines the encapsulation format, and specifies the VLAN identifier.
Step 4	`Router(config-if)# ip address` *ip-address mask*	Sets a primary IP address for an interface.

You can configure TRISL to route source-routed traffic by enabling the collection and use of RIF information on a TRISL subinterface. This creates a "pseudoring" to terminate the RIF path on a ring. Without RIF information, a packet could not be bridged across a source-route bridged network connected to this interface.

To route source-routed traffic, use the following additional commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# multiring trcrf-vlan vlanid ring ring-number`	Creates a pseudoring to terminate the RIF and assigns it to a VLAN.
Step 2	`Router(config-if)# multiring {protocol-keyword [all-routes | spanning | all | other]}`	Enables collection and use of RIF information with routed protocols.

Note TRISL encapsulation must be specified for a subinterface before an IP address can be assigned to that subinterface.

Configuring Hot Standby Router Protocol over TRISL

The Hot Standby Router Protocol (HSRP) provides fault tolerance and enhanced routing performance for IP networks. HSRP allows Cisco routers to monitor each other's operational status and very quickly assume packet forwarding responsibility in the event the current forwarding device in the HSRP group fails or is taken down for maintenance. The standby mechanism remains transparent to the attached hosts and can be deployed on any LAN type. With multiple hot-standby groups, routers can simultaneously provide redundant backup and perform load-sharing across different IP subsets.

To configure HSRP over TRISL between VLANs, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type slot/port subinterface-number`	Specifies the subinterface on which ISL will be used.
Step 2	`Router(config-if)# encapsulation tr-isl trbrf-vlan vlanid bridge-num bridge-number`	Defines the encapsulation format, and specify the VLAN identifier.
Step 3	`Router(config-if)# ip address ip-address mask [secondary]`	Specifies the IP address for the subnet on which ISL will be used.
Step 4	`Router(config-if)# standby [group-number] ip [ip-address [secondary]]`	Enables HSRP.

To customize hot standby group attributes, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# standby [group-number] timers hellotime holdtime`	Configures the time between hello packets and the hold time before other routers declare the active router to be down.
`Router(config-if)# standby [group-number] priority priority`	Sets the hot standby priority used to choose the active router.
`Router(config-if)# standby [group-number] preempt`	Specifies that if the local router has priority over the current active router, the local router should attempt to take its place as the active router.
`Router(config-if)# standby [group-number] track type-number [interface-priority].`	Configures the interface to track other interfaces, so that if one of the other interfaces goes down, the hot standby priority for the device is lowered.
`Router(config-if)# standby [group-number] authentication string`	Enables the automatic spanning-tree function on a group of bridged interfaces.

Configuring IPX Routing over TRISL

The IPX Routing over ISL VLANs feature extends Novell NetWare routing capabilities to include support for routing all standard IPX encapsulations for Token Ring frame types in VLAN configurations. Users with Novell NetWare environments can configure either SAP or SNAP encapsulations to be routed using the TRISL encapsulation across VLAN boundaries.

Netware users can now configure consolidated VLAN routing over a single VLAN trunking interface. With configurable Token Ring encapsulation protocols on a per VLAN basis, users have the flexibility of using VLANs regardless of their NetWare Token Ring encapsulation. Encapsulation types and corresponding framing types are described in the "Configuring Novell IPX" chapter of the *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

Note Only one type of IPX encapsulation can be configured per VLAN (subinterface). The IPX encapsulation used must be the same within any particular subnet. A single encapsulation must be used by all NetWare systems that belong to the same LAN.

To configure Cisco IOS software to route IPX on a router with connected VLANs, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing globally.
Step 2	`Router(config)# interface type slot/port.subinterface-number`	Specifies the subinterface on which TRISL will be used.
Step 3	`Router(config-if)# encapsulation tr-isl trbrf-vlan vlanid bridge-num bridge-number`	Defines the encapsulation for TRISL.
Step 4	`Router(config-if)# ipx encapsulation encapsulation-type`	Specifies the IPX encapsulation.
Step 5	`Router(config-if)# ipx network network number`	Specifies the IPX network.

Note The default IPX encapsulation format for Token Ring in Cisco IOS routers is SAP. Therefore, you only need to explicitly configure the IPX encapsulation type if your Token Ring network requires SNAP encapsulation instead of SAP.

When routing source-routed traffic for specific VLANs, use the following additional commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# multiring trcrf-vlan vlanid trcrf-ring ring-number`	Creates a pseudoring to terminate the RIF and assign it to a VLAN.
Step 2	`Router(config-if)# multiring {protocol-keyword [all-routes | spanning | all | other]}`	Enables collection and use of RIF information with routed protocols.

Configuring Source-Route Bridging over TRISL

To configure SRB over TRISL, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group vring-num`	Configures a virtual ring for the router.
Step 2	`Router(config)# interface type slot/port.subinterface-number`	Specifies the subinterface on which TRISL will be used.
Step 3	`Router(config-if)# encapsulation tr-isl trbrf-vlan vlanid bridge-num bridge-number`	Defines the encapsulation for TRISL.
Step 4	`Router(config-if)# source-bridge trcrf-vlan vlanid ring-group ring-number`	Attaches a TrCRF VLAN identifier to the router's virtual ring.

Configuring Source-Route Transparent Bridging over TRISL

To configure transparent bridging over TRISL, use the following command beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type slot/port.subinterface-number`	Specifies the subinterface on which TRISL will be used.
Step 2	`Router(config-if)# encapsulation tr-isl trbrf-vlan vlanid bridge-num bridge-number`	Defines the encapsulation for TRISL.
Step 3	`Router(config-if)# bridge-group bridge-group number`	Specifies the bridge group to which the TRISL subinterface belongs.

Configuring Source-Route Translational Bridging over TRISL

To configure source-route translational bridging over TRISL, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group vring-num`	Configures a virtual ring for the router.
Step 2	`Router(config)# source-bridge transparent ring-group pseudoring bridge-number tb-group [oui]`	Enables bridging between transparent bridging and source-route bridging.
Step 3	`Router(config)# interface type slot/port.subinterface-number`	Specifies the subinterface on which TRISL will be used.
Step 4	`Router(config-if)# encapsulation tr-isl trbrf-vlan vlanid bridge-num bridge-number`	Defines the encapsulation for TRISL.
Step 5	`Router(config-if)# source-bridge trcrf-vlan vlanid ring-group ring-number`	Assigns a VLAN ID to the router's virtual ring.

Note For a complete description of SR/TLB, including configuring translation compatibility with IBM 8209 bridges and configuring Token Ring LLC2 to Ethernet Type II (0x80d5) and Token Ring LLC2 to Ethernet 802.3 LLC2 (standard) translations, please refer to the "Configuring Source-Route Bridging" chapter in this publication and "Source-Route Bridging Commands" chapter in the *Cisco IOS Bridging and IBM Command Reference* (Volume 1 of 2).

Configuring Automatic Spanning Tree

The automatic spanning-tree function supports automatic resolution of spanning trees in SRB networks, which provides a single path for spanning explorer frames to traverse from a given node in the network to another. Spanning explorer frames have a single-route broadcast indicator set in the routing information field. Port identifiers consist of ring numbers and bridge numbers associated with the ports. The spanning-tree algorithm for SRB does not support Topology Change Notification Bridge Protocol Data Unit (BPDU).

Although the automatic spanning-tree function can be configured with Source-Route Translational Bridging (SR/TLB), the SRB domain and transparent bridging domain have separate spanning trees. Each Token Ring interface can belong to only one spanning tree. Only one bridge group can run the automatic spanning-tree function at a time.

To create a bridge group that runs an automatic spanning-tree function compatible with the IBM SRB spanning-tree implementation, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group protocol ibm`	Creates a bridge group that runs the automatic spanning-tree function.

To enable the automatic spanning-tree function for a specified group of bridged interfaces in SRB or SR/TLB, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge spanning` *bridge-group*	Enables the automatic spanning-tree function on a group of bridged interfaces.

Monitoring TRISL Statistics

You can collect, clear, and display statistical information about the network.

The Duplicate Ring Protocol (DRiP) runs on Cisco routers and switches that support switched VLAN networking and is used to identify active Token Ring VLANs (TrCRFs).

DRiP maintains the status of TrCRFs and uses this information to determine whether there are multiple TrCRFs active in a TrBRF.

DRiP information is used for the following:

- All-routes explorer filtering

  DRiP information is used in conjunction with the local configuration to determine which of the TrCRFs configured within a TrBRF have active ports. This information is used on the base switch to correctly filter all-routes explorers and on the ISL module to discard AREs that have already been on an attached ring.

- Detecting the configuration of duplicate TrCRFs across routers and switches, which would cause a TrCRF to be distributed across ISL trunks

  DRiP information is used in conjunction with the local configuration information to determine which TrCRFs are already active on the switches. If a TrCRF is enabled on more than one switch or router, the ports associated with the TrCRF are disabled on all switches. A router will not disable the internal ring used for SRB and for routing source-routed traffic. Instead, the router generates the following error message to indicate that two identical TrCRFs exist:

```
DRIP conflict with CRF <vlan-id>
```

To show or clear DRiP or VLAN statistics, use one or all the following command in privileged EXEC mode:

Command	Purpose
`Router# clear drip counters`	Clears DRiP counters.
`Router# clear vlan statistics`	Removes VLAN statistics from any statically configured or system configured entries.
`Router# show drip`	Displays DRiP information.
`Router# show vlans`	Displays a summary of VLAN subinterfaces.

Note When DRiP counters are cleared, the counter is reset to 0. Incrementing of DRiP counters indicates that the router is receiving packets across the TrBRF.

TRISL Configuration Examples

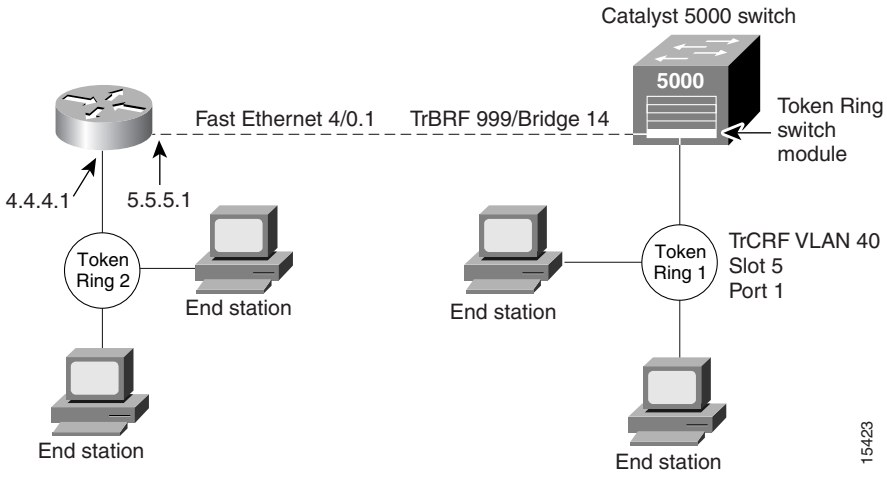
The following sections provide TRISL configuration examples:

Note Because the VLAN Trunk Protocol (VTP) is not supported on the router configured with TRISL, the TrCRF configuration on the router must also be specified in the Catalyst 5000 switch configuration.

IP Routing Non-Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface Example

Figure 55 illustrates IP routing between a TRISL VLAN and a Token Ring interface.

Figure 55 IP Routing Between a TRISL VLAN and a Token Ring Interface



The following is the configuration for the router:

```
ip routing
interface TokenRing 3/1
 ip address 4.4.4.1 255.255.255.0
!
interface fastethernet4/0.1
 ip address 5.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf 999 bridge-num 14
```

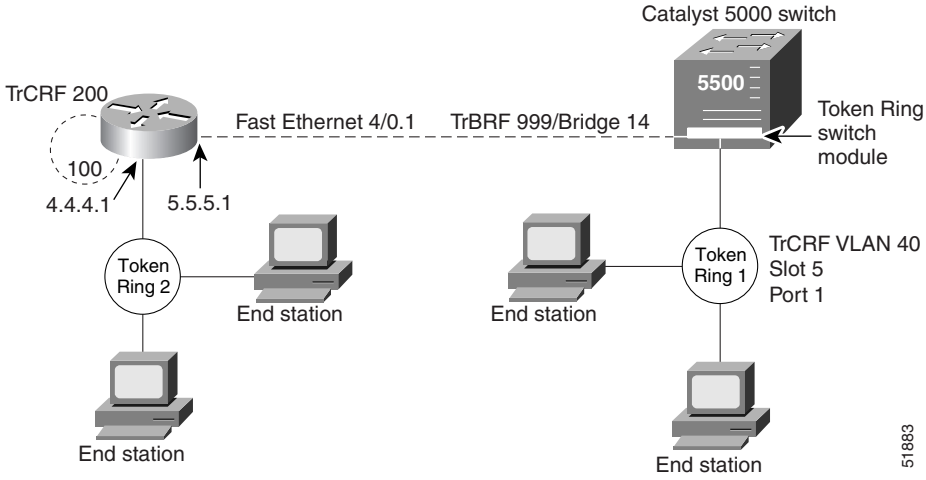
The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port 1 is assigned to the TrCRF VLAN 40.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ieee
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srt
#add token port to trcrf 40
set vlan 40   5/1
set trunk 1/2 on
```

IP Routing Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface Example

Figure 56 illustrates IP routing source-routed frames between a TRISL VLAN and a Token Ring interface.

Figure 56 Routing Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface



The following is the configuration for the router:

```
ip routing
interface TokenRing 3/1
 ip address 4.4.4.1 255.255.255.0
!
interface fastethernet4/0.1
 ip address 5.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
```

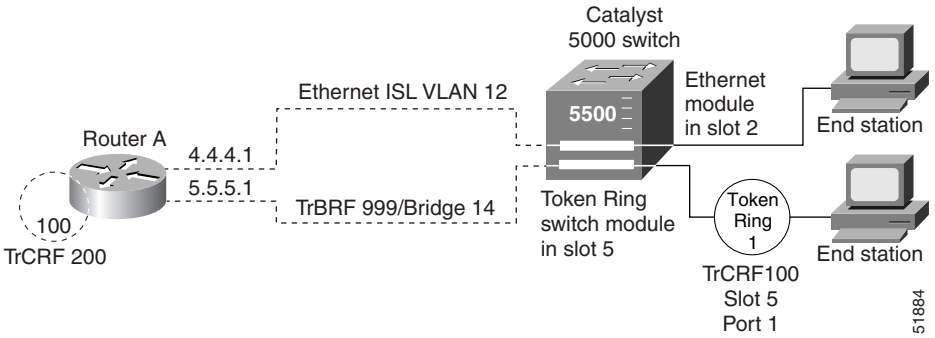
The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned to the TrCRF VLAN 40.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srb
#add token port to trcrf 40
set vlan 40   5/1
set trunk 1/2 on
```

IP Routing Source-Route Frames Between a TRISL VLAN and an Ethernet ISL VLAN Example

Figure 57 illustrates IP routing source-route frames between a TRISL VLAN and an Ethernet ISL VLAN.

Figure 57 IP Routing Source-Routed Frames Between a TRISL VLAN and an Ethernet ISL VLAN



The following is the configuration for the router:

```
interface fastethernet4/0.1
 ip address 5.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
!
interface fastethernet4/0.2
 ip address 4.4.4.1 255.255.255.0
 encapsulation isl 12
```

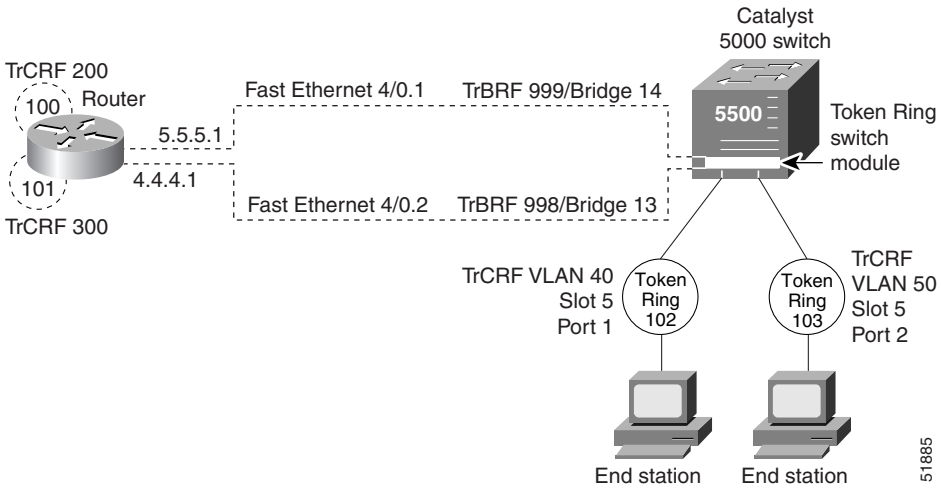
The following is the configuration for the Catalyst 5000 switch with the Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port is assigned with TrCRF VLAN 100 and the Ethernet port is assigned with VLAN 12.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 100 name trcrf100 type trcrf parent 999 ring 0x1 mode srb
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 12 name eisl12 type ethernet
#add token port to trcrf 100
set vlan 100 5/1
#add ethernet
set vlan 12 2/1
   set trunk 1/2 on
```

IP Routing Source-Routed Frames Between TRISL VLANs Example

Figure 58 illustrates IP routing source-routed frames between two TrBRF VLANs.

Figure 58 IP Routing Source-Routed Frames Between TrBRF VLANs



The following is the configuration for the router:

```
interface fastethernet4/0.1
 ip address 5.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
!
interface fastethernet4/0.2
 ip address 4.4.4.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 998 bridge-num 13
 multiring trcrf-vlan 300 ring 101
 multiring all
```

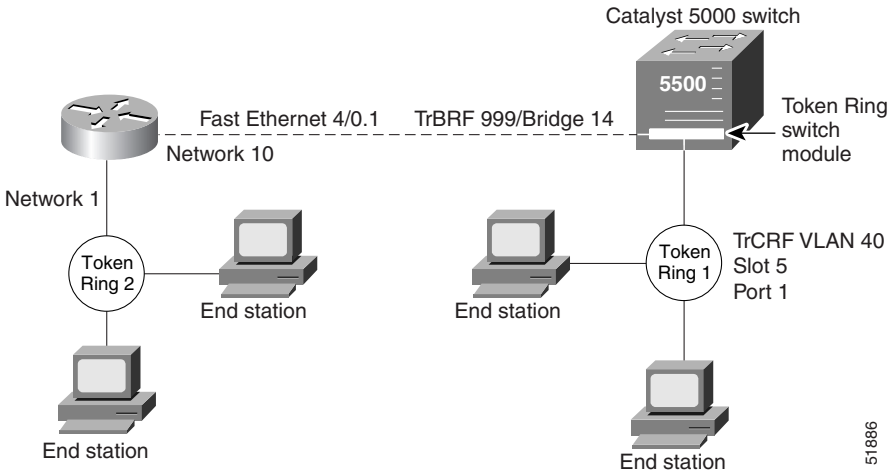
The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port attached to ring 102 is assigned with TrCRF VLAN 40 and the Token Ring port attached to ring 103 is assigned with TrCRF VLAN 50.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x66 mode srb
set vlan 998 name trbrf type trbrf bridge 0xd stp ibm
set vlan 300 name trcrf300 type trcrf parent 998 ring 0x65 mode srb
set vlan 50 name trcrf50 type trcrf parent 998 ring 0x67 mode srb
#add token port to trcrf 40
set vlan 40   5/1
#add token port to trcrf 50
set vlan 50   5/2
set trunk 1/2 on
```

IPX Routing Non-Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface Example

Figure 59 shows IPX routing non-source-routed frames between a TRISL VLAN and a Token Ring interface.

Figure 59 IPX Routing Non-Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface Example



The following is the configuration for the router:

```
ipx routing
interface TokenRing 3/1
 ipx network 1
!
interface fastethernet4/0.1
 ipx network 10
 encapsulation tr-isl trbrf 999 bridge-num 14
```

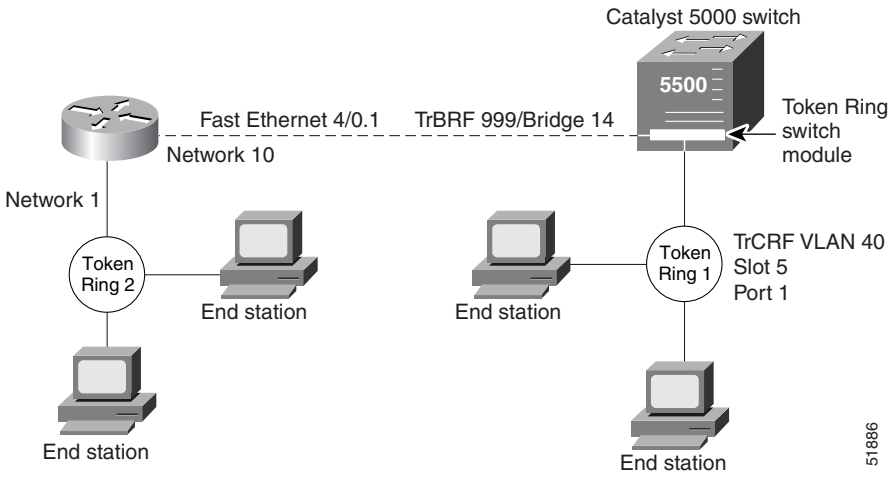
The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port attached to ring 1 is assigned to the TrCRF VLAN 40.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ieee
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srt
#add token port to trcrf 40
set vlan 40   5/1
set trunk 1/2 on
```

IPX Routing Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface Example

Figure 60 shows IPX routing source-routed frames between a TRISL VLAN and a Token Ring interface.

Figure 60 IPX Routing Source-Routed Frames Between a TRISL VLAN and a Token Ring Interface



The following is the configuration for the router:

```
ipx routing
!
interface TokenRing 3/1
 ipx network 1
 multiring all
!
interface fastethernet4/0.1
 ipx network 10
 encapsulation tr-isl trbrf 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
```

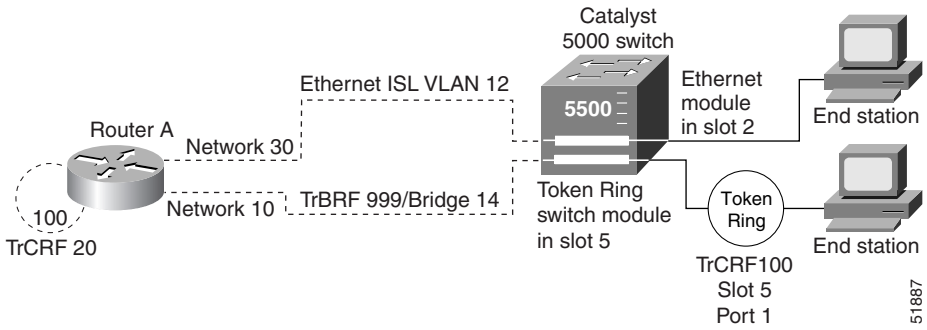
The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port attached to ring 1 is assigned to the TrCRF VLAN 40.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srb
#add token port to trcrf 40
set vlan 40   5/1
set trunk 1/2 on
```

IPX Routing Source-Route Frames Between a TRISL VLAN and an Ethernet ISL VLAN Example

Figure 61 shows IPX routing source-route frames between a TRISL VLAN and an Ethernet ISL VLAN.

Figure 61 IPX Routing Source-Routed Frames Between a TRISL VLAN and an Ethernet ISL VLAN



The following is the configuration for the router:

```
ipx routing
interface fastethernet4/0.1
 ipx network 10
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 20 ring 100
 multiring all
!
interface fastethernet4/0.2
 ipx network 30
 encapsulation isl 12
```

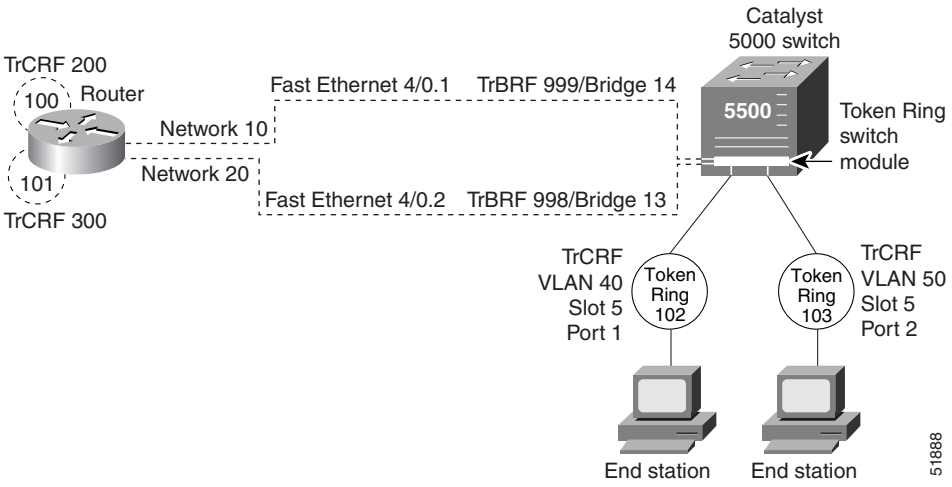
The following is the configuration for the Catalyst 5000 switch with the Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port is assigned with TrCRF VLAN 100 and the Ethernet port is assigned with VLAN 12.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 100 name trcrf100 type trcrf parent 999 ring 0x1 mode srb
set vlan 20 name trcrf20 type trcrf parent 999 ring 0x64 mode srb
set vlan 12 name default type eisl12
#add token port to trcrf 100
set vlan 100 5/1
#add ethernet
set vlan 12 2/1
set trunk 1/2 on
```

IPX Routing Source-Routed Frames Between TRISL VLANs Example

Figure 62 shows IPX source-routed frames between TRISL VLANs.

Figure 62 IPX Routing Source-Routed Frames Between TRISL VLANs



The following is the configuration for the router:

```
ipx routing
interface fastethernet4/0.1
 ipx network 10
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
!
interface fastethernet4/0.2
 ipx network 20
 encapsulation tr-isl trbrf-vlan 998 bridge-num 13
 multiring trcrf-vlan 300 ring 101
 multiring all
```

The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port attached to ring 102 is assigned with TrCRF VLAN 40 and the Token Ring port attached to ring 103 is assigned with TrCRF VLAN 50.

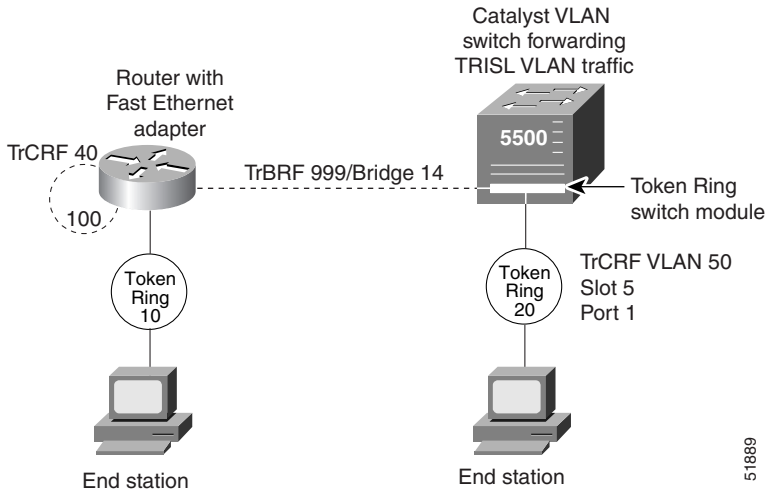
```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x66 mode srb
set vlan 998 name trbrf type trbrf bridge 0xd stp ibm
set vlan 300 name trcrf300 type trcrf parent 998 ring 0x65 mode srb
set vlan 50 name trcrf50 type trcrf parent 998 ring 0x67 mode srb
#add token port to trcrf 40
set vlan 40   5/1
#add token port to trcrf 50
set vlan 50   5/2
```

```
set trunk 1/2 on
```

SRB Between Token Ring and TRISL VLAN Example

Figure 63 illustrates SRB between a Token Ring interface on a router and a TRISL VLAN.

Figure 63 SRB Between a Token Ring Interface and TRISL VLAN



The following is the configuration for the router with the Token Ring interface:

```
source-bridge ring-group 100
!
interface TokenRing3/1
 ring speed 16
 source-bridge 10 1 100
 source-bridge spanning
!
interface fastethernet4/0.1
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 source-bridge trcrf-vlan 40 ring-group 100
 source-bridge spanning
!
```

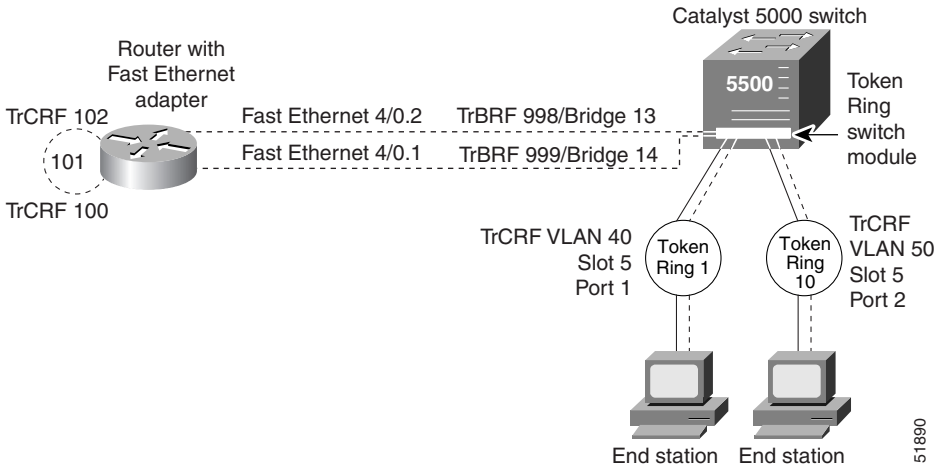
The following is the configuration for the Catalyst 5000 switch:

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x64 mode srb
set vlan 50 name trcrf50 type trcrf parent 999 ring 0x14 mode srb
#add token port to trcrf 50
set vlan 50   5/1
```

SRB Between TRISL VLANs Example

Figure 64 illustrates SRB between two TrCRF VLANs.

Figure 64 SRB Between TRISL VLANs



The following is the configuration for the router:

```
source-bridge ring-group 101
!
interface fastethernet4/0.1
 encapsulation tr-isl trbrf 999 bridge-num 14
 source-bridge trcrf-vlan 100 ring-group 101
 source-bridge spanning
!
interface fastethernet4/0.2
 encapsulation tr-isl trbrf 998 bridge-num 13
 source-bridge trcrf-vlan 102 ring-group 101
 source-bridge spanning
```

The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. The Token Ring port on 5/1 is assigned to TrCRF VLAN 40 and the Token Ring port on 5/2 is assigned to TrCRF VLAN 50.

In this configuration, the keyword *name* is optional and *srb* is the default mode.

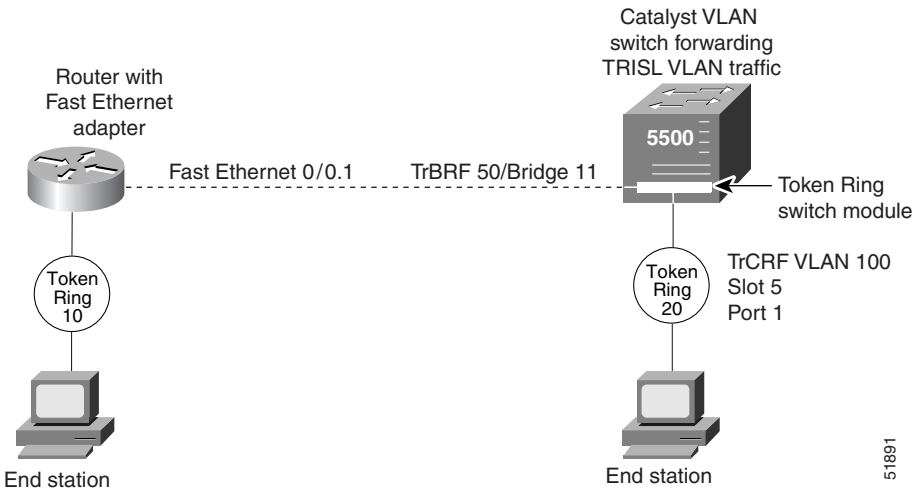
```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 100 name trcrf100 type trcrf parent 999 ring 0x65 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srb
set vlan 998 name trbrf type trbrf bridge 0xd stp ibm
set vlan 102 name trcrf102 type trcrf parent 998 ring 0x65 mode srb
set vlan 50 name trcrf50 type trcrf parent 998 ring 0xa mode srb
#add token port to trcrf 40
set vlan 40   5/1
#add token port to trcrf 50
set vlan 50   5/2
```

```
#enable trunk
set trunk 1/2 on
```

Transparent Bridging Between Token Ring and TRISL VLAN Example

Figure 65 illustrates transparent bridging between a router's Token Ring interface and a TRISL VLAN.

Figure 65 Transparent Bridging Between Token Ring and TRISL VLAN



The following is the configuration for the router:

```
bridge 1 protocol ieee
!
interface Tokenring0
 bridge-group 1
!
interface fastethernet0/0.1
 encapsulation tr-isl trbrf-vlan 50 bridge-num 11
 bridge-group 1
```

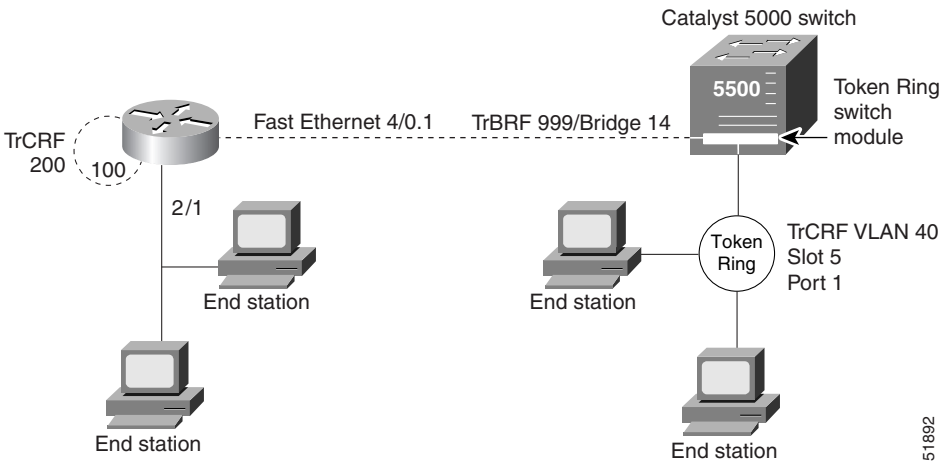
The following is the configuration for the Catalyst 5000 switch with a Token Ring switch module in slot 5:

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 50 name trbrf50 type trbrf bridge 0xb stp ieee
set vlan 100 name trcrf100 type trcrf ring 0x14 parent 50 mode srt
#enable trunk
set trunk 1/2 on
#add token port to trcrf 100
set vlan 100 5/1
```

SR/TLB Between a TRISL VLAN and an Ethernet Interface Example

Figure 66 illustrates SR/TLB between a TRISL VLAN and an Ethernet interface.

Figure 66 SR/TLB Between a TRISL VLAN and an Ethernet Interface



The following is the configuration for the router:

```
source-bridge ring-group 100
source-bridge transparent 100 101 6 1
!
interface Ethernet2/0
 bridge-group 1
!
interface fastethernet4/0.1
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 source-bridge trcrf-vlan 200 ring-group 100
 source-bridge spanning
!
bridge 1 protocol ieee
!
```

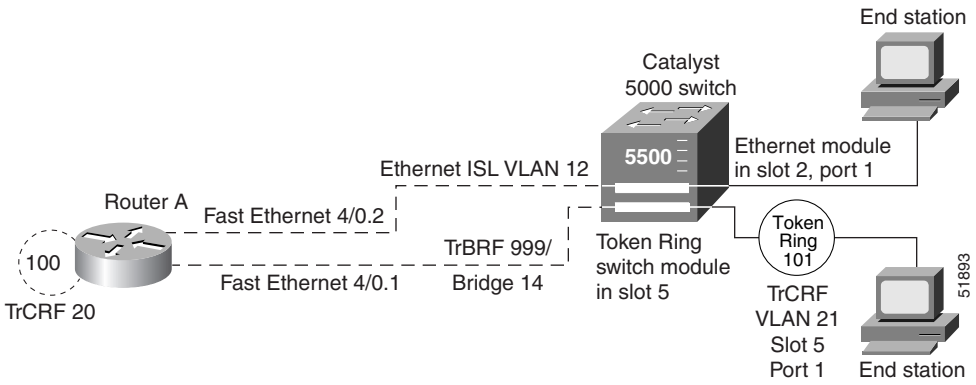
The following is the configuration for the Catalyst 5000 switch with an Ethernet card in module 5 and using port 1. The Token Ring port on 5/1 is assigned to TrCRF VLAN 40.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf999 type trbrf bridge 0xe stp ibm
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srb
#add token port to trcrf 40

set vlan 40   5/1
#enable trunk
set trunk 1/2 on
```

SR/TLB Between a TRISL VLAN and an Ethernet ISL VLAN Example

Figure 67 illustrates SR/TLB between a TRISL VLAN and an Ethernet ISL VLAN.

Figure 67 SR/TLB Between a TRISL VLAN and an Ethernet ISL VLAN



The following is the configuration for the router:

```
source-bridge ring-group 100
source-bridge transparent 100 101 6 1
!
interface fastethernet4/0.1
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 source-bridge trcrf-vlan 20 ring-group 100
 source-bridge spanning
!
interface fastethernet4/0.2
 encapsulation isl 12
 bridge-group 1
!
bridge 1 protocol ieee
```

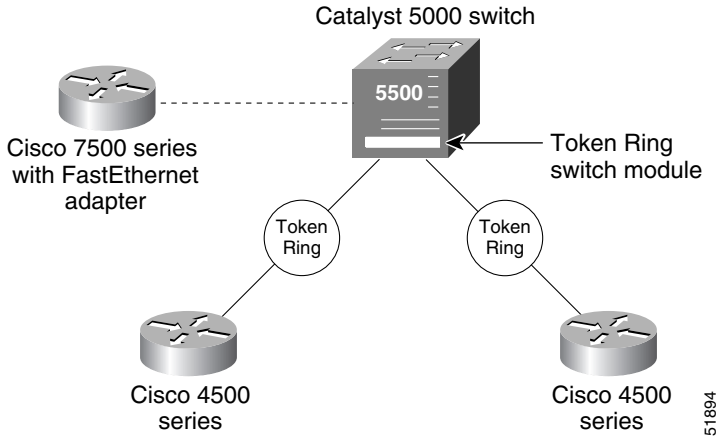
The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port attached to ring 101 is assigned to TrCRF VLAN 21, and the router's virtual ring is assigned to TrCRF VLAN 20.

```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 type trbrf bridge 0xe stp ibm
set vlan 20 type trcrf parent 999 ring 0x64 mode srb
set vlan 21 type trcrf parent 999 ring 0x65 mode srb
#add token port to trcrf 21
set vlan 21 5/1
#add ethernet
set vlan 12 type ethernet
set vlan 12 2/1
set trunk 1/2 on
```

TRISL with Fast EtherChannel Example

Figure 68 illustrates TRISL with Fast EtherChannel.

Figure 68 Sample Configuration of TRISL with Fast EtherChannel



The following is the configuration for the Cisco 7500:

```
source-bridge ring-group 50
interface Port-channel1
  no ip address
  no ip directed-broadcast
  no ip route-cache distributed
  hold-queue 300 in
interface Port-channel1.1
  encapsulation tr-isl trbrf-vlan 20 bridge-num 1
  ip address 10.131.25.1 255.255.255.0
  no ip redirects
  no ip directed-broadcast
  source-bridge trcrf-vlan 23 ring-group 50
  source-bridge spanning
interface Port-channel1.2
  encapsulation tr-isl trbrf-vlan 30 bridge-num 2
  ip address 10.131.24.1 255.255.255.0
  no ip redirects
  no ip directed-broadcast
  source-bridge trcrf-vlan 33 ring-group 50
  source-bridge spanning
interface fastethernet4/1/0
  no ip address
  no ip directed-broadcast
  no ip route-cache distributed
  channel-group 1
interface fastethernet4/1/1
  no ip address
  no ip directed-broadcast
  no ip route-cache distributed
  channel-group 1
```

The following is the configuration for the Catalyst 5000 Switch:

```
 set vlan 10 name VLAN0010 type ethernet mtu 1500 said 100010 state active
 set vlan 20 name VLAN0020 type trbrf mtu 4472 said 100020 state active bridge 0x1 stp
ieee
```

```
 set vlan 30 name VLAN0030 type trbrf mtu 4472 said 100030 state active bridge 0x2 stp
ieee

 set vlan 22 name VLAN0022 type trcrf mtu 4472 said 100022 state active parent 20 ring 0x1
mode srt aremaxhop 7 stemaxhop 7
 set vlan 23 name VLAN0023 type trcrf mtu 4472 said 100023 state active parent 20 ring
0x32 mode srt aremaxhop 7 stemaxhop 7
 set vlan 32 name VLAN0032 type trcrf mtu 4472 said 100032 state active parent 30 ring 0x2
mode srt aremaxhop 7 stemaxhop 7
 set vlan 33 name VLAN0033 type trcrf mtu 4472 said 100033 state active parent 30 ring
0x32 mode srt aremaxhop 7 stemaxhop 7

 set port channel 1/1-2 on

 set trunk 1/1  on isl 1-1005
 set trunk 1/2  on isl 1-1005
add token port to crf 22
set vlan 22 5/1
add token port to crf 32
set vlan 32 5/2
```

TRISL with Fast EtherChannel only runs on the Cisco 7500. The MTU size can be set to more than 1500 if all the members of the port channel interface are 2FE/ISL adaptors. If, on the other hand, any member of the port channel interface is a non 2FE/ISL adaptor, then the MTU size is not configurable and defaults to 1500 bytes. Also, only IP utilizes all four links. Spanning Tree Protocol must be disabled if transparent bridging is configured on the FEC. The port-channel interface is the routed interface. Do not enable Layer 3 addresses on the physical Fast Ethernet interfaces. Do not assign bridge groups on the physical Fast Ethernet interfaces because it creates loops. Also, you must disable Spanning Tree Protocol if transparent bridging is configured on the FEC.

Configuring Token Ring Route Switch Module

This chapter describes how to configure Token Ring virtual LANs (VLANs) on the route switch module (RSM). For a complete description of the commands mentioned in this chapter, refer to the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

The Token Ring RSM feature is supported on the RSM in the Catalyst 5000 platform. Support for the Token Ring RSM feature was first introduced in the Cisco IOS Release 11.3(5)T. The Token Ring RSM feature is supported on all RSM Cisco IOS Release 12.0 T images. A list of the supported Cisco IOS releases and software images are located in the *Release Notes for Catalyst 5000 Family RSM/VIP2 Cisco IOS 12.0 T Software Releases* publication. A complete description of the RSM can be found in the *Catalyst 5000 Family Route Switch Module Installation and Configuration Note* and the *Route Switch Module Catalyst VIP2-15 and VIP2-40 Installation and Configuration Note.*

The Token Ring VLAN support on the RSM adds the capability to do multiprotocol routing and bridging for Token Ring VLANs on the RSM. The RSM is a router module running Cisco IOS software that plugs into a switch that supports Token Ring. This section provides a brief overview of Token Ring switching.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

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

Technology Overview

The term switching was originally used to describe packet-switch technologies such as Link Access Procedure, Balanced (LAPB), Frame Relay, Switched Multimegabit Data Service (SMDS), and X.25. Today, LAN switching refers to a technology that is similar to a bridge in many ways.

Like bridges, switches connect LAN segments and use information contained in the frame to determine the segment to which a datagram needs to be sent. Switches, however, operate at much higher speeds than bridges, and can support new functionality, such as VLANs. See the "VLAN" section on page 3 and the "Token Ring VLANs" section on page 3.

Token Ring switches first appeared in 1994. The first-generation Token Ring switches can be divided into two basic categories:

- Processor-based switches—These switches use reduced instruction set computer (RISC) processors to switch Token Ring frames. Although they typically have a lot of function, they are slow and relatively expensive. These switches have been deployed mainly as backbone switches because of their high cost.
- Application-specific integrated circuit (ASIC)-based switches with limited functionality—These switches are fast and relatively inexpensive, but have very limited function. Typically, they offer little to no filtering, limited management information, limited support for bridging modes, and limited VLANs. Today, although these switches are less expensive than processor-based switches, they are still too expensive and limited for widespread use of dedicated Token Ring to the desktop.

In 1997, a second generation of Token Ring switches was introduced. Cisco's second-generation Token Ring switches use ASIC-based switching, but they provide increased functionality resulting in a higher speed and lower cost. They also provide a wider variety of function than their predecessors, including support for multiple bridging modes, Dedicated Token Ring (DTR) on all ports, high-port density, high-speed links, filtering, Remote Monitoring (RMON) management, broadcast control, and flexible VLANs.

The family of second-generation Token Ring switches can be used for backbone switching, workgroup microsegmentation, and dedicated Token Ring to the desktop. Switches currently being offered that support Token Ring include:

- The Catalyst 3900, which is a stackable workgroup Token Ring switch that provides support for all switching modes, filtering, RMON, DTR, and SNMP management, and support for Asynchronous Transmission Mode (ATM) and Inter-Switch Link (ISL).
- The Catalyst 3920, which is a also a stackable workgroup Token Ring switch that provides support for all switching modes, filtering, RMON, DTR, and SNMP management.
- The Catalyst 5000, which is a modular switch that supports Ethernet, Fast Ethernet, Fiber Distributed Data Interface (FDDI), ATM, and now Token Ring.

The Catalyst Token Ring switches support the following bridging modes: source-route bridging (SRB), source-route transparent bridging (SRT), and source-route switching.

Usability of Switching

The traditional method of connecting multiple Token Ring segments is to use a SRB. For example, bridges are often used to link workgroup rings to the backbone ring. However, the introduction of the bridge can significantly reduce performance at the user's workstation. Further problems might be introduced by aggregate traffic loading on the backbone ring.

To maintain performance and avoid overloading the backbone ring, you can locate servers on the same ring as the workgroup that needs to access the server. However, dispersing the servers throughout the network makes them more difficult to back up, administer, and secure than if they are located on the backbone ring. Dispersing the servers also limits the number of servers that particular stations can access.

Collapsed backbone routers might offer greater throughput than bridges, and can interconnect a larger number of rings without becoming overloaded. Routers provide both bridging and routing functions between rings and have sophisticated broadcast control mechanisms. These mechanisms become increasingly important as the number of devices on the network increases.

The main drawback of using routers as the campus backbone is the relatively high price per port and the fact that the throughput typically does not increase as ports are added. A Token Ring switch is designed to provide wire speed throughput regardless of the number of ports in the switch. In addition, the Catalyst 3900 Token Ring switch can be configured to provide very low latency between Token Ring ports by using cut-through switching.

As a local collapsed backbone device, a Token Ring switch offers a lower per-port cost and can incur lower interstation latency than a router. In addition, the switch can be used to directly attach large numbers of clients or servers, thereby replacing concentrators. Typically, a Token Ring switch is used in conjunction with a router, providing a high-capacity interconnection between Token Ring segments while retaining the broadcast control and wide-area connectivity provided by the router.

VLAN

A VLAN is a logical group of LAN segments, independent of physical location, with a common set of requirements. For example, several end stations might be grouped as a department, such as engineering or accounting. If the end stations are located close to one another, they can be grouped into a LAN segment. If any of the end stations are on a different LAN segment, such as different buildings or locations, they can be grouped into a VLAN that has the same attributes as a LAN even though the end stations are not all on the same physical segment. The information identifying a packet as part of a specific VLAN is preserved across a Catalyst switch connection to a router or another switch if they are connected via trunk ports, such as ISL or ATM.

Token Ring VLANs

Because a VLAN is essentially a broadcast domain, a Token Ring VLAN is slightly more complex than an Ethernet VLAN. In transparent bridging, there is only one type of broadcast frame and, therefore, only one level of broadcast domain and one level of VLAN. In source routing, however, there are two types of broadcast frames:

- Those that are confined to a single ring
- Those that traverse the bridged domain

Therefore, there are two levels of VLANs in a Token Ring switched network. These two categories of broadcast frames result in a broadcast domain that is hierarchical in nature, just as a local ring domain can exist only within a domain of all the inter-connected rings.

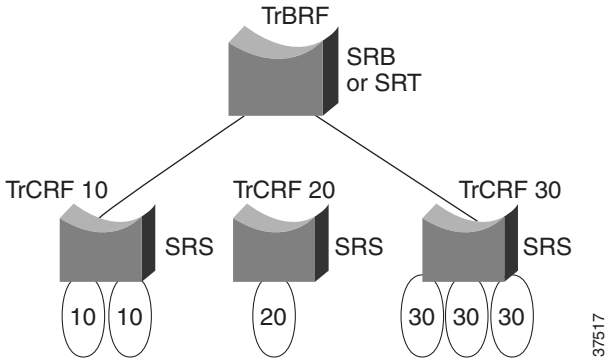
The first level is the Token Ring Concentrator Relay Function (TrCRF). In a Token Ring VLAN, logical ring domains are formed by defining groups of ports that have the same ring number. The IEEE calls such a port group a Concentrator Relay Function (CRF). On Catalyst switches, such a grouping of Token Ring ports is called a Token Ring CRF (TrCRF). At this level, the VLAN is a logical ring and, as such, is assigned a ring number. On a Token Ring switch, the logical ring (TrCRF) contains one or more physical ports. Source-route switching is used to forward frames within a TrCRF based on Media Access

Control (MAC) address or Route Descriptor. On an RSM, a logical ring (TrCRF) can be defined that does not contain any physical ports, but rather is used only in processing source-routed traffic to terminate the RIF.

The second level of VLAN is the Token Ring Bridge Relay Function (TrBRF). This is the parent VLAN to which TrCRF VLANs are assigned. The domain of interconnected rings is formed using an internal multiport bridge function that the IEEE calls a Bridge Relay Function (BRF). On Catalyst switches, such a grouping of logical rings is called a Token Ring BRF (TrBRF). At this level, the VLAN is a logical bridge and, as such, is assigned a bridge number. The TrBRF is responsible for forwarding frames between groups of ports with the same ring number (TrCRFs) via either SRB or SRT.

Figure 69 depicts the relationship between TrCRF and TrBRF VLANS.

Figure 69 Token Ring VLAN Support on the RSM



Token Ring VLAN Support on the RSM

The Token Ring VLAN support on the RSM adds the capability to do multi-protocol routing and bridging for Token Ring VLANs on the RSM. The RSM can be used alone to do inter-VLAN routing, or it can be paired with a Catalyst second-generation Versatile Interface Processor (VIP2) to provide external network connections with the same port adapters used on Cisco 7500 series routers. The RSM/VIP2 combination provides routing between VLANs and Catalyst VIP2 port adapters.

The Token Ring VLAN support on the RSM adds the following functionality to the Catalyst 5000 switch:

- IP routing for source-routed and non-source-routed frames between Token Ring (TrBRF) and/or Ethernet VLANs and VIP2 interfaces
- IPX routing for source-routed and non-source-routed frames between Token Ring (TrBRF) and/or Ethernet VLANs and VIP2 interfaces
- Source-route bridging SRB between Token Ring (TrBRF) VLANs and VIP2 interfaces
- Source-route translational bridging (SR/TLB) between Token Ring (TrBRF) VLANs and Ethernet VLANs and VIP2 interfaces
- Source-route transparent bridging (SRT) between Token Ring (TrBRF) VLANs and SRT-capable VLANs and VIP2 interfaces

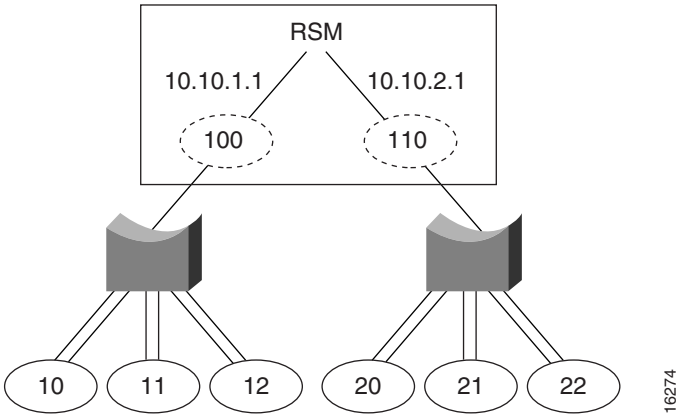
Both APPN and DLSw+ are supported for Token Ring VLANs on the RSM. However, RSRB is not supported on the RSM.

For information on how Token Ring VLANs are implemented on switches, refer to the *Catalyst Token Ring Switching Implementation Guide,* the *Catalyst 5000 Series Token Ring Configuration Notes,* the *Catalyst 3900 Token Ring Switching User Guide,* and the *Catalyst 3920 Token Ring Switching User Guide*.

The RSM is a router module running Cisco IOS router software that directly interfaces (plugs into) the Catalyst switch backplane. From the Token Ring VLAN perspective, the interface to the RSM is at the Token Ring bridged network (TrBRF) level. With the RSM, it is possible to route or bridge between separate Token Ring and Ethernet domains.

When routing or bridging between TrBRF VLANs that are defined as SRB domains, it is necessary to create a logical ring on the RSM for proper RIF processing. This logical ring is defined as a TrCRF VLAN that does not contain any external Token Ring switch ports. Figure 70 illustrates the logical view of IP routing between two source-route bridged VLANs on the RSM. In this view, the RSM appears to have an interface to both ring 100 and ring 110.

Figure 70 Logical View of VLAN Support on the RSM



Related Documents

For related information on this feature, see the following documents:

- *Cisco IOS Bridging and IBM Networking Configuration Guide*
- *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2)
- *Catalyst 5000 Series RSM Installation and Configuration Note*
- *Catalyst Token Ring Switching Implementation Guide*
- *Catalyst 5000 Series Token Ring Configuration Notes*
- *Catalyst 3900 Token Ring Switching User Guide*
- *Catalyst 3920 Token Ring Switching User Guide*.
- *Catalyst 5000 Family Route Switch Module Installation and Configuration Note*
- *Route Switch Module Catalyst VIP2-15*
- *Release Notes for Catalyst 5000 Family RSM/VIP2 Cisco IOS 12.0 T Software Releases*
- *VIP2-40 Installation and Configuration Note*

Prerequisites

Before you configure bridging or routing for Token Ring VLAN interfaces on the RSM, configure the VLANs on the Catalyst 5000 supervisor engine module.

To configure a VLAN on the supervisor engine module, use the following command in privileged EXEC mode:

Command	Purpose
Router# **set vlan** *vlan_num* [**name** *name*] [**type** {**ethernet** | **fddi** | **fddinet** | **trcrf** | **trbrf**}] [**state** {**active** | **suspend**}] [**said** *said*] [**mtu** *mtu*] [**ring** *hex_ring_number*] [**decring** *decimal_ring_number*] [**bridge** *bridge_num*] [**parent** *vlan_num*] [**mode** {**srt** | **srb**}] [**stp** {**ieee** | **ibm** | **auto**}] [**translation** *vlan_num*] [**backupcrf** {**off** | *on*}] [**aremaxhop** *hopcount*] [**stemaxhop** *hopcount*]	Configures a TrCRF or TrBRF on the supervisor engine module.

TRRSM Configuration Task List

To configure and monitor Token Ring VLAN support on the RSM, perform one or more of the following tasks:

- Configuring IP Routing, page 6
- Configuring IPX Routing, page 7
- Configuring Source-Route Bridging, page 8
- Configuring Source-Route Transparent Bridging, page 8
- Configuring Source-Route Translational Bridging, page 9
- Configuring Automatic Spanning Tree, page 9

See the "TRRSM Configuration Examples" section on page 11 for examples.

Note For information on configuring DLSw+, refer to the "Configuring Data-Link Switching Plus" chapter in this publication and the "DLSw+ Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Configuring IP Routing

The IP routing for Token Ring VLANs feature extends IP routing capabilities to include support for routing IP frame types in Token Ring VLAN configurations. IP can be routed between:

- TrBRFs
- TrBRFs and the VIP2
- TrBRFs and Ethernet VLANs

To configure IP routing on an RSM, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip routing`	Enables IP routing on the RSM.
Step 2	`Router(config)# interface vlan vlanid type [trbrf | ethernet]`	Configures a Token Ring or Ethernet interface on the RSM.
Step 3	`Router(config-if)# ip address ip-address mask`	Sets a primary IP address for an interface.

You can configure an RSM to route source-routed traffic by creating a "pseudoring" to terminate the RIF path on a ring and by enabling the collection and use of RIF information.

To route source-routed traffic, use the following additional commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# multiring trcrf-vlan vlanid ring ring-number`	Creates a pseudoring to terminate the RIF and assigns it to a VLAN.
Step 2	`Router(config-if)# multiring {protocol-keyword [all-routes | spanning | all | other]}`	Enables collection and use of RIF information with routed protocols.

Configuring IPX Routing

The IPX routing support for Token Ring VLANs extends Novell NetWare routing capabilities to include support for routing IPX on Token Ring VLAN interfaces and on Ethernet VLAN interfaces. IPX can be routed between:

- TrBRFs
- TrBRFs and the VIP2
- TrBRFs and Ethernet VLANs

Users with Novell NetWare environments can configure either SAP or SNAP encapsulations to be routed across VLAN boundaries.

To configure Cisco IOS software to route IPX on an RSM with connected Token Ring VLANs, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing globally.
Step 2	`Router(config)# interface vlan vlanid type [trbrf | ethernet]`	Configures a Token Ring or Ethernet interface on the RSM.
Step 3	`Router(config-if)# ipx encapsulation encapsulation-type`	Specifies the IPX encapsulation.
Step 4	`Router(config-if)# ipx network network number`	Specifies the IPX network.

Note The default IPX encapsulation format for Token Ring in the Cisco IOS software is SAP. Therefore, you only need to explicitly configure the IPX encapsulation type if your Token Ring network requires SNAP encapsulation instead of SAP.

When routing source-routed traffic for specific VLANs, use the following additional commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# multiring trcrf-vlan vlanid ring ring-number`	Creates a pseudoring to terminate the RIF and assigns it to a VLAN.
Step 2	`Router(config-if)# multiring {protocol-keyword [all-routes | spanning | all | other]}`	Enables collection and use of RIF information with routed protocols.

Configuring Source-Route Bridging

To configure SRB on the RSM, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group vring-num`	Configures a virtual ring for the RSM.
Step 2	`Router(config)# interface vlan vlanid type [trbrf | ethernet]`	Configures a Token Ring or Ethernet interface on the RSM.
Step 3	`Router(config-if)# source-bridge trcrf-vlan vlanid ring-group ring-number`	Attaches a TrCRF VLAN identifier to the RSM's virtual ring.

Configuring Source-Route Transparent Bridging

To configure SRT on the RSM, use the following command beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface vlan vlanid type [trbrf | ethernet]`	Configures a Token Ring or Ethernet interface on the RSM.
Step 2	`Router(config-if)# bridge-group bridge-group number`	Specifies the bridge group to which the interface belongs.

Configuring Source-Route Translational Bridging

To configure SR/TLB on the RSM, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group vring-num`	Configures a virtual ring for the RSM.
Step 2	`Router(config)# source-bridge transparent ring-group pseudoring bridge-number tb-group [oui]`	Enables bridging between transparent bridging and source-route bridging.
Step 3	`Router(config)# interface vlan vlanid type [trbrf | ethernet]`	Configures a Token Ring or Ethernet interface on the RSM.
Step 4	`Router(config-if)# source-bridge trcrf-vlan vlanid ring-group ring-number`	Assigns a VLAN ID to the RSM's virtual ring.

Note For a complete description of SR/TLB, including configuring translation compatibility with IBM 8209 bridges and configuring Token Ring LLC2 to Ethernet Type II (0x80d5) and Token Ring LLC2 to Ethernet 802.3 LLC2 (standard) translations, refer to the "Configuring Source-Route Bridging" chapter in this publication and the "Source-Route Bridging Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Configuring Automatic Spanning Tree

The automatic spanning-tree function supports automatic resolution of spanning trees in SRB networks, which provides a single path for spanning explorer frames to traverse from a given node in the network to another. Spanning explorer frames have a single-route broadcast indicator set in the RIF. Port identifiers consist of ring numbers and bridge numbers associated with the ports. The spanning-tree algorithm for SRB does not support Topology Change Notification Bridge Protocol Data Unit (BPDU).

Although the automatic spanning-tree function can be configured with SR/TLB, the SRB domain and transparent bridging domain have separate spanning trees. Each Token Ring interface can belong to only one spanning tree. Only one bridge group can run the automatic spanning-tree function at a time.

To create a bridge group that runs an automatic spanning-tree function compatible with the IBM SRB spanning-tree implementation, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge bridge-group protocol ibm`	Creates a bridge group that runs the automatic spanning-tree function.

To enable the automatic spanning-tree function for a specified group of bridged interfaces in SRB or SR/TLB, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge spanning bridge-group`	Enables the automatic spanning-tree function on a group of bridged interfaces.

Verifying TRRSM

To verify that Token Ring Route Switch Module (TRRSM) is configured, use the following command in privileged EXEC mode:

Command	Purpose
`Router# show running configuration`	Displays the running configuration of a device.

The following output indicates this is a Token Ring VLAN because of the trbrf interface:

```
milan# show run
version 12.0
!
hostname Sample
!
interface Vlan61 type trbrf
 no ip address
 no ip directed-broadcast
 no ip route-cache
 ethernet-transit-oui 90-compatible
```

To verify the status of the Token Ring VLAN, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# show ip interfaces brief`	Lists a summary of an interface's IP information and status
`Router# show interface`	Displays statistics for all interfaces configured on the router or access server

The following output indicates that Vlan5 and Vlan61 interfaces are up:

```
RSM-A# show ip interface brief
Interface IP-Address OK? Method Status Protocol

Vlan5           unassigned YES unset   up                    up
Vlan61      unassigned YES NVRAM   up                up
```

The following output shows the statistics for Vlan61:

```
RSM-A# show interface vlan61
Vlan61 is up, line protocol is up
  Hardware is Cat5k Virtual Token Ring, address is 0009.d49e.0100 (bia
0009.d49e.0100)
  MTU 4464 bytes, BW 16000 Kb, DLY 630 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation SNAP, loopback not set
  ARP type:SNAP, ARP Timeout 04:00:00
  Ring speed:16 Mbps
  Duplex:half
  Mode:Classic token ring station
```

Monitoring Statistics

You can collect, display, and clear statistical information about the network. The Duplicate Ring Protocol (DRiP) runs on Cisco routers and switches that support switched VLAN networking and is used to identify active Token Ring VLANs (TrCRFs). DRiP maintains the status of TrCRFs and uses this information to determine whether there are multiple TrCRFs active in a TrBRF.

DRiP information is used for the following:

- All-routes explorer filtering

  DRiP information is used in conjunction with the local configuration to determine which of the TrCRFs configured within a TrBRF have active ports. This information is used on the base switch to correctly filter all-routes explorers and on the RSM to discard AREs that have already been on an attached ring.

- Detecting the configuration of duplicate TrCRFs across routers and switches

  DRiP information is used in conjunction with the local configuration information to determine which TrCRFs are already active on the switches. If a TrCRF is enabled on more than one switch or router, the ports associated with the TrCRF are disabled on all switches. The RSM will not disable the internal ring used for processing source-routed traffic. Instead, the RSM generates the following error message to indicate that two identical TrCRFs exist:

  ```
  DRIP conflict with CRF <vlan-id>
  ```

To show or clear DRiP or VLAN statistics, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# clear drip counters`	Clears DRiP counters.
`Router# show drip`	Displays DRiP information.

Note When DRiP counters are cleared, the counter is reset to 0. Incrementing of DRiP counters indicates that the router is receiving packets across the TrBRF.

TRRSM Configuration Examples

The following sections provide the following RSM and Catalyst 5000 switch configuration examples:

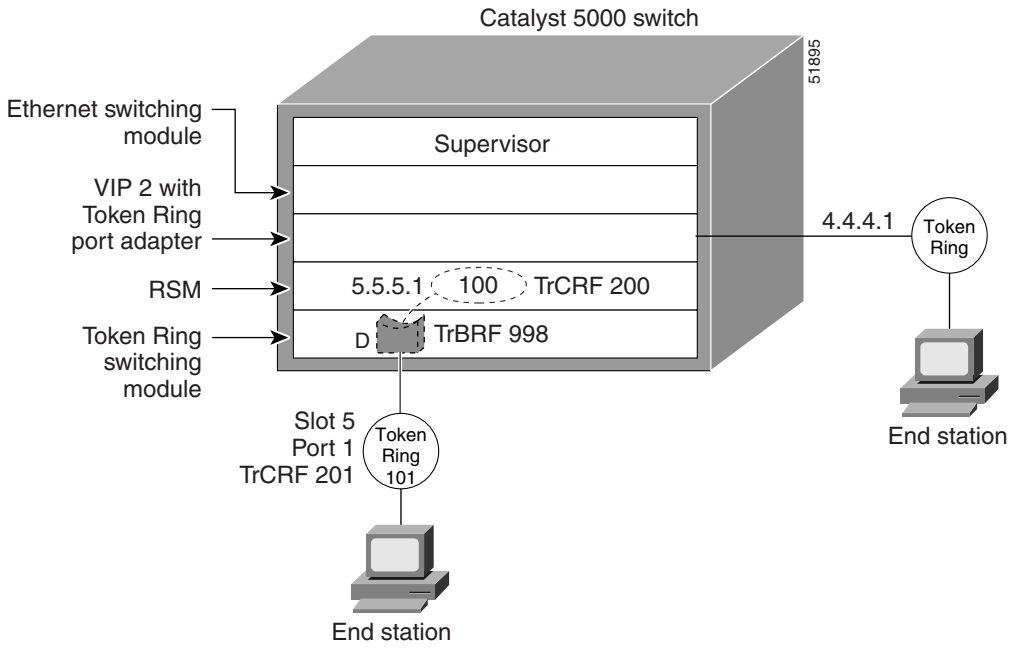
- IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface Example, page 12
- IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Ethernet Interface Example, page 13
- IP Routing Source-Routed Frames Between TrBRF VLANs Example, page 14
- IP Routing Source-Routed Frames Between a TrBRF VLAN and an Ethernet VLAN Example, page 15
- IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface Example, page 16

IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface Example

Following is the configuration for the RSM as shown in Figure 71:

```
interface TokenRing 3/1
 ip address 4.4.4.1 255.255.255.0
 multiring all
!
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
 multiring trcrf-vlan 200 ring 100
 multiring ip
```

Figure 71 IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with the TrCRF VLAN 201:

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ibm
set vlan 200 type trcrf parent 998 ring 0x64 mode srb
set vlan 201 type trcrf parent 998 ring 0x65 mode srb
#add token port to trcrf 201
set vlan 201 5/1
```

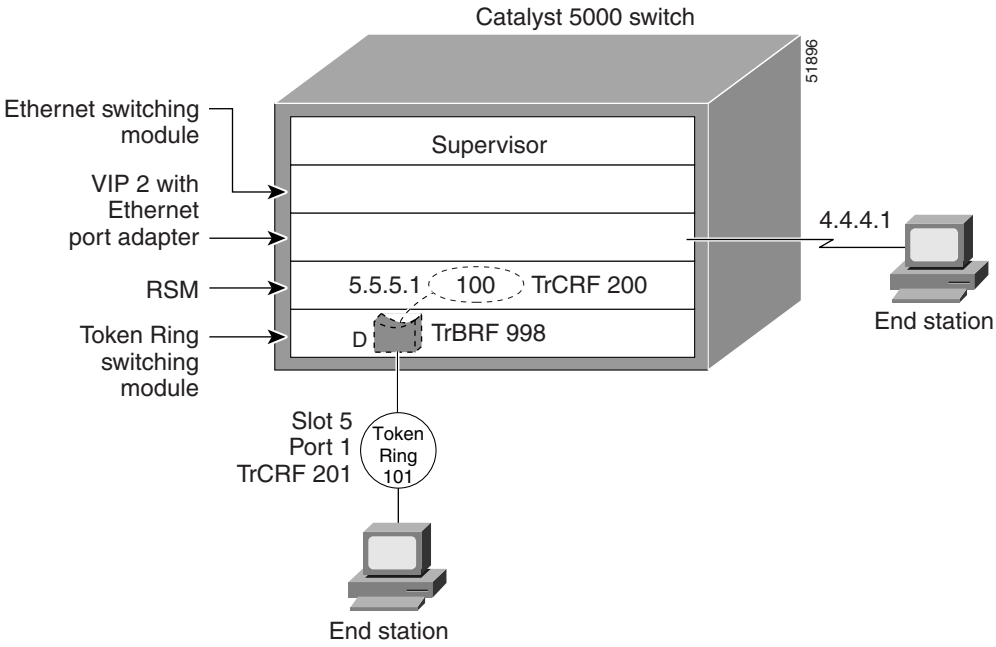
Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Ethernet Interface Example

Following is the configuration for the RSM as shown in Figure 72:

```
interface Ethernet 2/2
 ip address 4.4.4.1 255.255.255.0
!
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
 multiring trcrf-vlan 200 ring 100
 multiring all
```

Figure 72 IP Routing Source-Routed Frames Between a TrBRF VLAN and a VIP2 Ethernet Interface



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with TrCRF VLAN 201.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ibm
set vlan 200 type trcrf parent 998 ring 0x64 mode srb
set vlan 201 type trcrf parent 998 ring 0x65 mode srb
#add token ring port to trcrf 201
set vlan 201 5/1
```

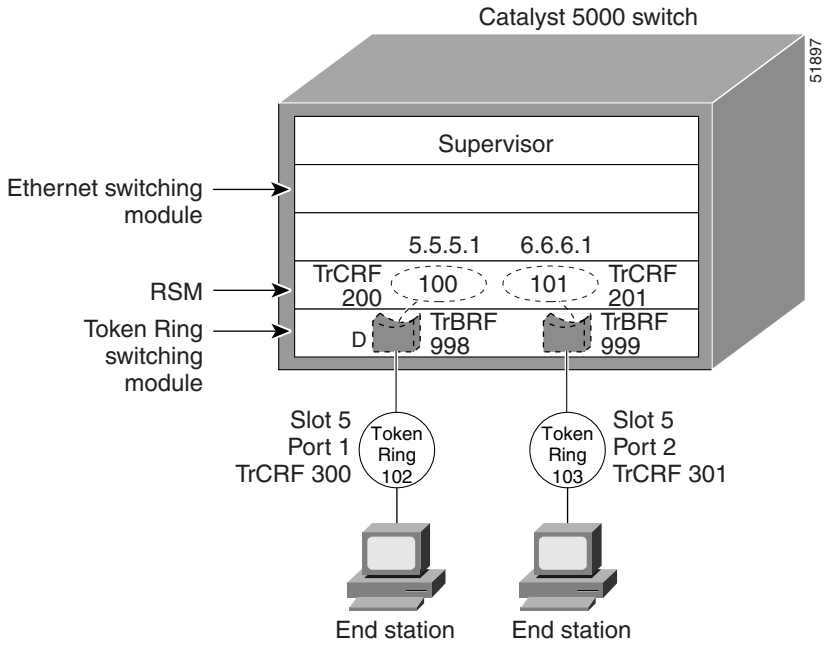
Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

IP Routing Source-Routed Frames Between TrBRF VLANs Example

Following is the configuration for the RSM as shown in Figure 73:

```
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
 multiring trcrf-vlan 200 ring 100
 multiring all
interface vlan999 type trbrf
 ip address 6.6.6.1 255.255.255.0
 multiring trcrf-vlan 201 ring 101
 multiring all
```

Figure 73 IP Routing Source-Routed Frames Between TrBRF VLANs



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with TrCRF VLAN 300 and the Token Ring port 5/2 is assigned with TrCRF VLAN 301.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ibm
set vlan 200 type trcrf parent 998 ring 0x64 mode srb
set vlan 300 type trcrf parent 998 ring 0x66 mode srb
set vlan 999 type trbrf bridge 0xe stp ibm
set vlan 201 type trcrf parent 999 ring 0x65 mode srb
set vlan 301 type trcrf parent 999 ring 0x67 mode srb
#add token port to trcrfs
set vlan 300 5/1
set vlan 301 5/2
```

Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

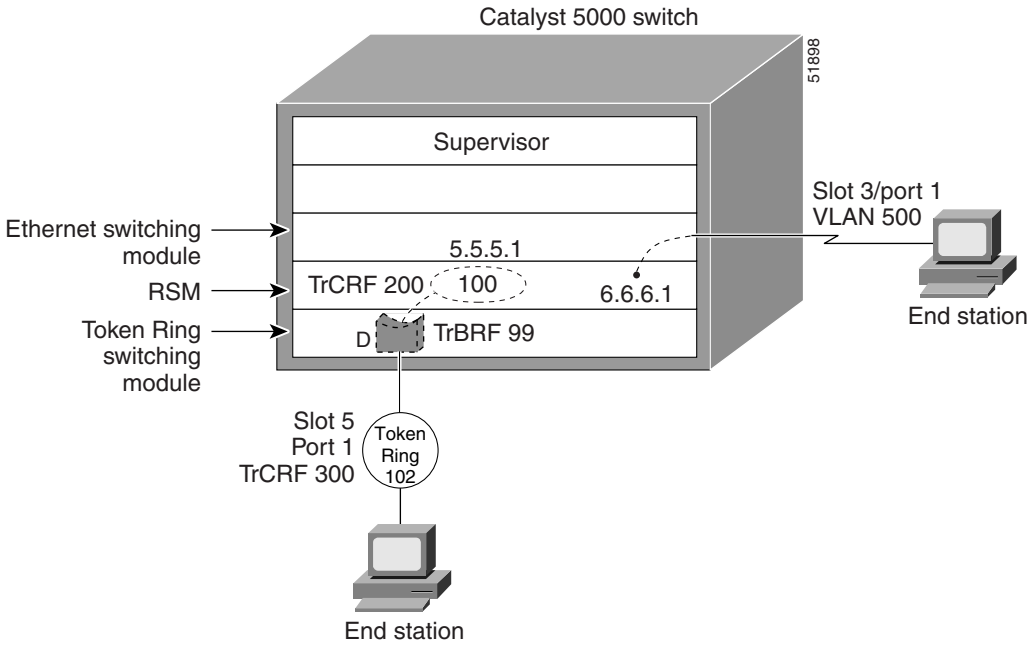
IP Routing Source-Routed Frames Between a TrBRF VLAN and an Ethernet VLAN Example

Following is the configuration for the RSM as shown in Figure 74:

```
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
 multiring trcrf-vlan 200 ring 100
```

```
 multiring all
interface vlan500 type ethernet
 ip address 6.6.6.1 255.255.255.0
```

Figure 74 IP Routing Source-Routed Frames Between a TrBRF VLAN and an Ethernet VLAN



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 3 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with TrCRF VLAN 300 and the Ethernet port 3/1 is assigned with VLAN 500.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ibm
set vlan 200 type trcrf parent 998 ring 0x64 mode srb
set vlan 300 type trcrf parent 998 ring 0x66 mode srb
#add token port to trcrf 300
set vlan 300 5/1
#add ether port to 500
set vlan 500 3/1
```

Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface Example

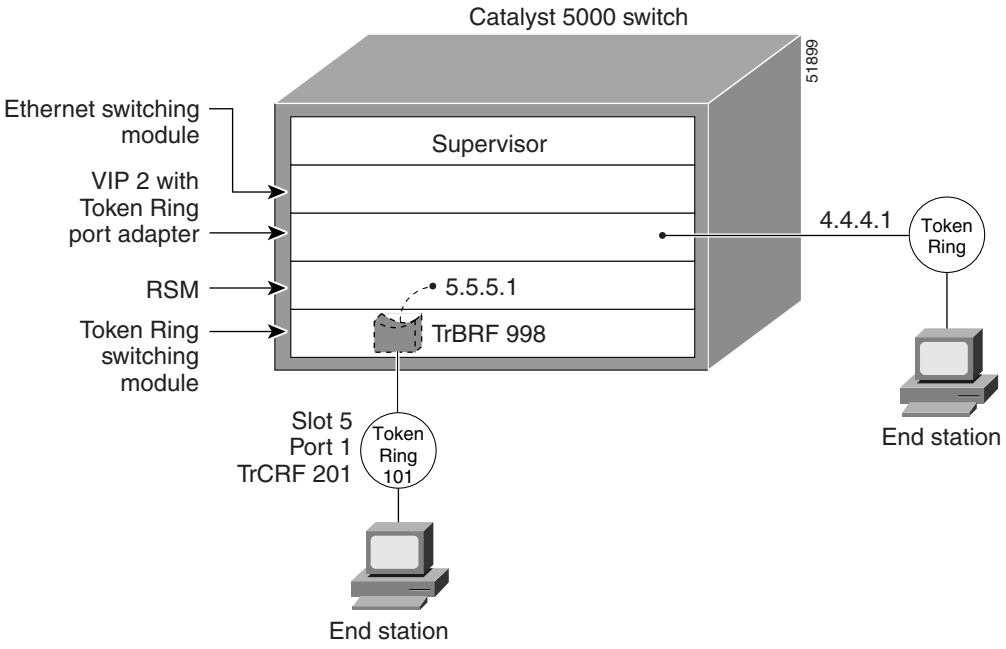
Following is the configuration for the RSM as shown in Figure 75:

```
interface TokenRing 3/1
```

```
 ip address 4.4.4.1 255.255.255.0
!
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
```

Note The **multiring** command is not needed in this configuration because these are non-source-routed frames.

Figure 75 IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and a VIP2 Token Ring Interface



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with the TrCRF VLAN 201.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ieee
set vlan 201 type trcrf parent 998 ring 0x65 mode srt
#add token port to trcrf 201
set vlan 201 5/1
```

Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

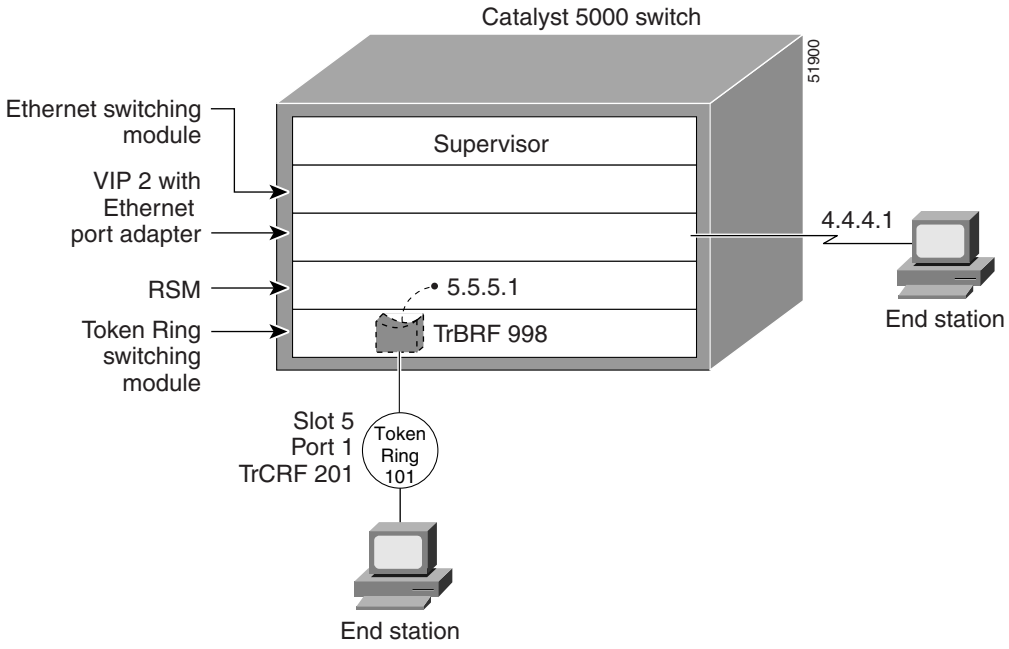
IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and a VIP2 Ethernet Interface Example

Following is the configuration for the RSM as shown in Figure 76:

```
interface Ethernet 2/2
 ip address 4.4.4.1 255.255.255.0
!
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
```

Note The **multiring** command is not needed in this configuration because these are non-source-routed frames.

Figure 76 IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and a VIP2 Ethernet Interface



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with TrCRF VLAN 201.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ieee
set vlan 201 type trcrf parent 998 ring 0x65 mode srt
#add token ring port to trcrf 201
set vlan 201 5/1
```

Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

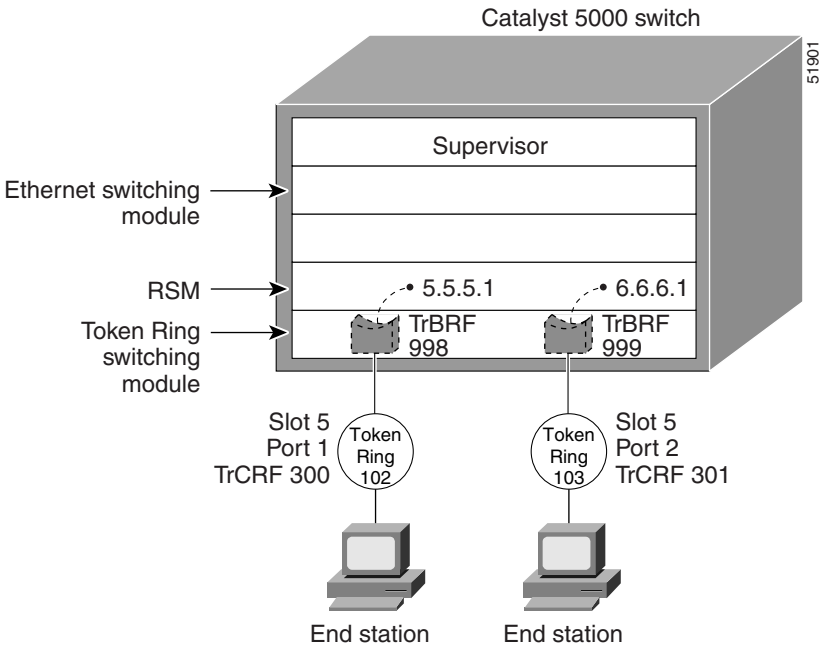
IP Routing Non-Source-Routed Frames Between TrBRF VLANs Example

Following is the configuration for the RSM as shown in Figure 77:

```
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
interface vlan999 type trbrf
 ip address 6.6.6.1 255.255.255.0
```

Note The **multiring** command is not needed in this configuration because these are non-source-routed frames.

Figure 77 IP Routing Non-Source-Routed Frames Between TrBRF VLANs



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with VLAN 300 and the Token Ring port 5/2 is assigned with VLAN 301.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ieee
set vlan 300 type trcrf parent 998 ring 0x66 mode srt
set vlan 999 type trbrf bridge 0xd stp ieee
set vlan 301 type trcrf parent 999 ring 0x67 mode srt
#add token port to trcrfs
set vlan 300 5/1
```

```
set vlan 301 5/2
```

Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

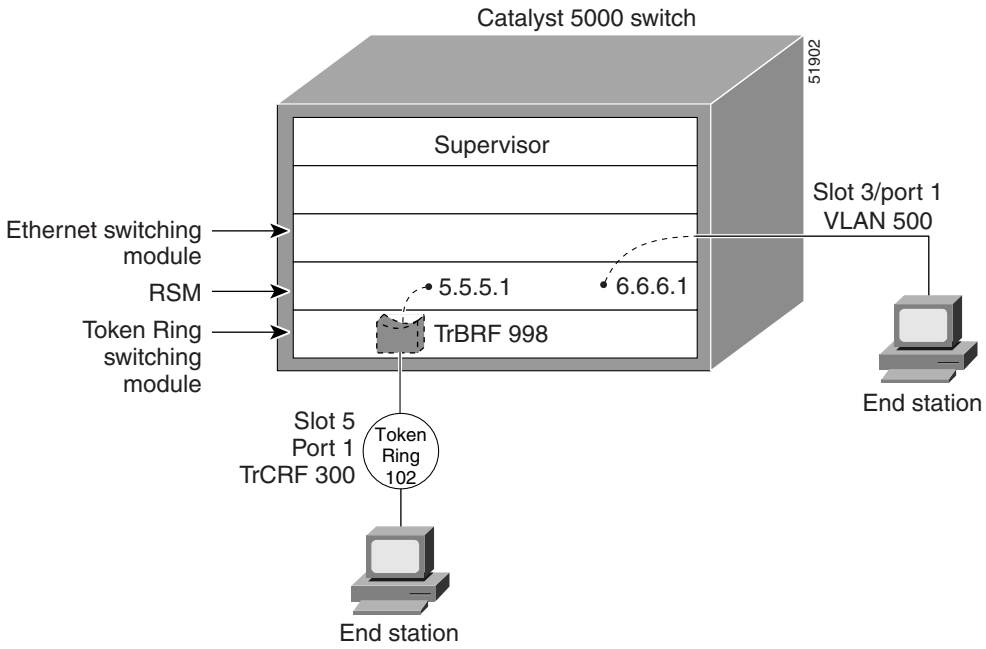
IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and an Ethernet VLAN Example

Following is the configuration for the RSM as shown in Figure 78:

```
interface vlan998 type trbrf
 ip address 5.5.5.1 255.255.255.0
!
interface vlan500 type ethernet
 ip address 6.6.6.1 255.255.255.0
```

Note The **multiring** command is not needed in this configuration because these are not source-routed frames.

Figure 78 IP Routing Non-Source-Routed Frames Between a TrBRF VLAN and an Ethernet VLAN



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 3 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with TrCRF VLAN 300 and the Ethernet port 3/1 is assigned with VLAN 500.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 998 type trbrf bridge 0xd stp ieee
```

```
set vlan 300 type trcrf parent 998 ring 0x66 mode srt
#add token port to trcrf 300
set vlan 300 5/1
#add ether port to 500
set vlan 500 3/1
```

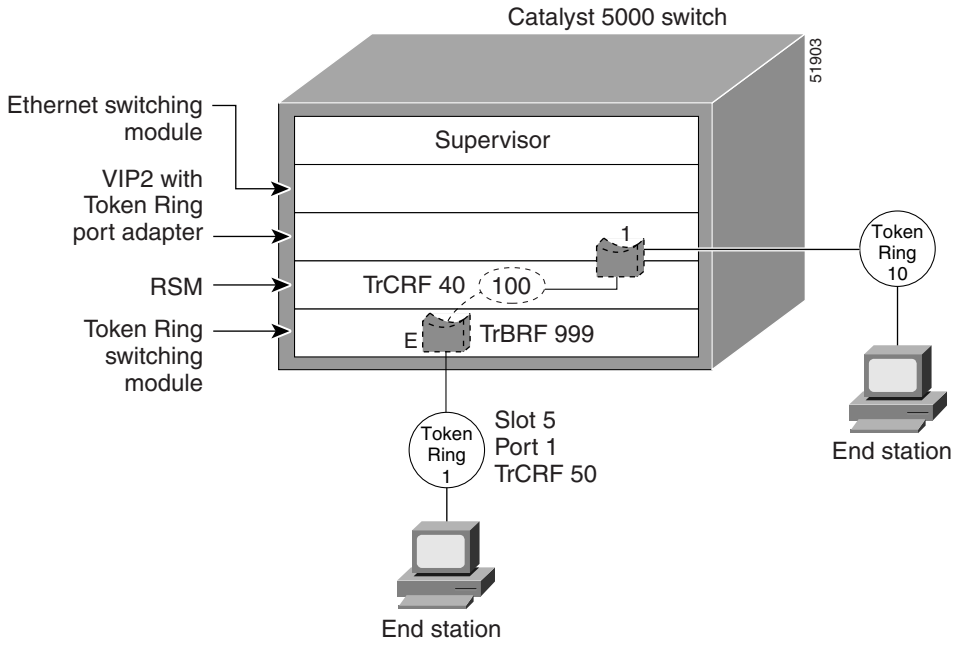
Because the VLANs are defined on a per-port basis, end stations connected to ports that belong to the same VLAN should belong to the same IP subnetwork.

SRB Between a TrBRF VLAN and VIP2 Token Ring Interface Example

The following configuration refers to the RSM as shown in Figure 79:

```
interface TokenRing3/1
  source-bridge 10 1 100
  source-bridge spanning
!
source-bridge ring-group 100
interface vlan999 type trbrf
  source-bridge trcrf-vlan 40 ring-group 100
  source-bridge spanning
```

Figure 79 SRB Between a TrBRF VLAN and VIP2 Token Ring Interface



Following is the configuration for the Catalyst 5000 switch which performs simple SRB. In this configuration, the Token Ring switch module exists in slot 5 and is using port 1. The Token Ring port on 5/1 is assigned to TrCRF VLAN 50.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
```

```
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x64 mode srb
set vlan 50 name trcrf50 type trcrf parent 999 ring 0x1 mode srb
#add token port to trcrf 50
set vlan 50 5/1
```

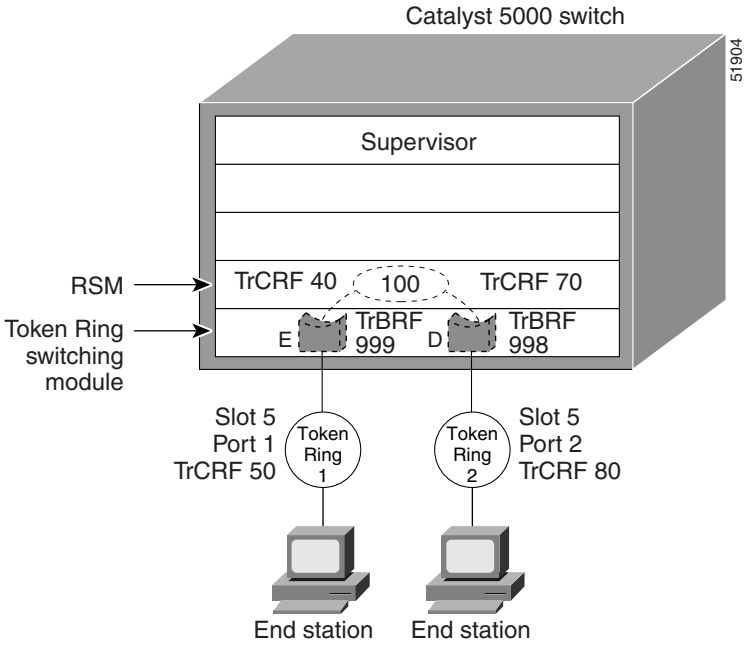
In this configuration, the keyword **name** is optional and *srb* is the default mode. The ring number on the switch must be specified in decimal by substituting the **decring** keyword for ring.

SRB Between TrBRF VLANs Example

The following configuration refers to the RSM as shown in Figure 80:

```
source-bridge ring-group 100
!
interface vlan999 type trbrf
  source-bridge trcrf-vlan 40 ring-group 100
  source-bridge spanning
!
interface vlan998 type trbrf
  source-bridge trcrf-vlan 70 ring-group 100
  source-bridge spanning
```

Figure 80 SRB Between TrBRF VLANs



The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. The Token Ring port on 5/1 is assigned to TrCRF VLAN 50 and the Token Ring port on 5/2 is assigned to TrCRF VLAN 80.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
```

```
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ibm
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x64 mode srb
set vlan 50 name trcrf50 type trcrf parent 999 ring 0x1 mode srb
set vlan 998 name trbrf type trbrf bridge 0xd stp ibm
set vlan 70 name trcrf70 type trcrf parent 998 ring 0x64 mode srb
set vlan 80 name trcrf80 type trcrf parent 998 ring 0x2 mode srb
#add token port to trcrf 50
set vlan 50 5/1
#add token port to trcrf 80
set vlan 80 5/2
```

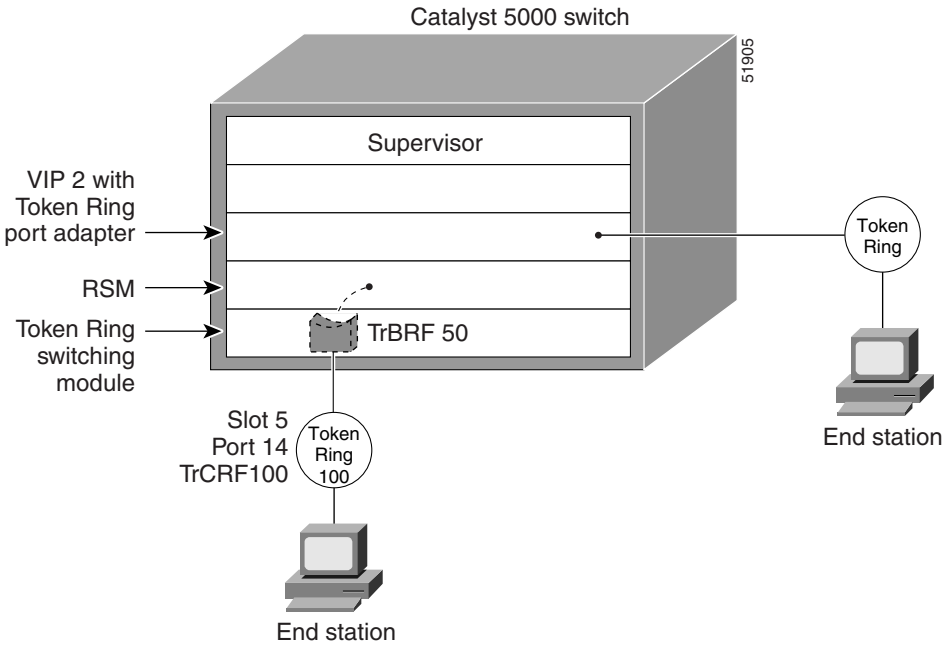
In this configuration, the keyword *name* is optional and *srb* is the default mode.

Transparent Bridging Between a TrBRF VLAN and a VIP2 Token Ring Interface Example

The following is the configuration for the RSM as shown in Figure 81:

```
bridge 1 protocol ieee
!
interface tokenring0
 bridge-group 1
interface vlan 50 type trbrf
 bridge-group 1
```

Figure 81 Transparent Bridging Between a TrBRF VLAN and a VIP2 Token Ring Interface



The following is the configuration for the Catalyst 5000 switch with a Token Ring switch module in slot 5:

```
#vtp
set vtp domain trrsm
```

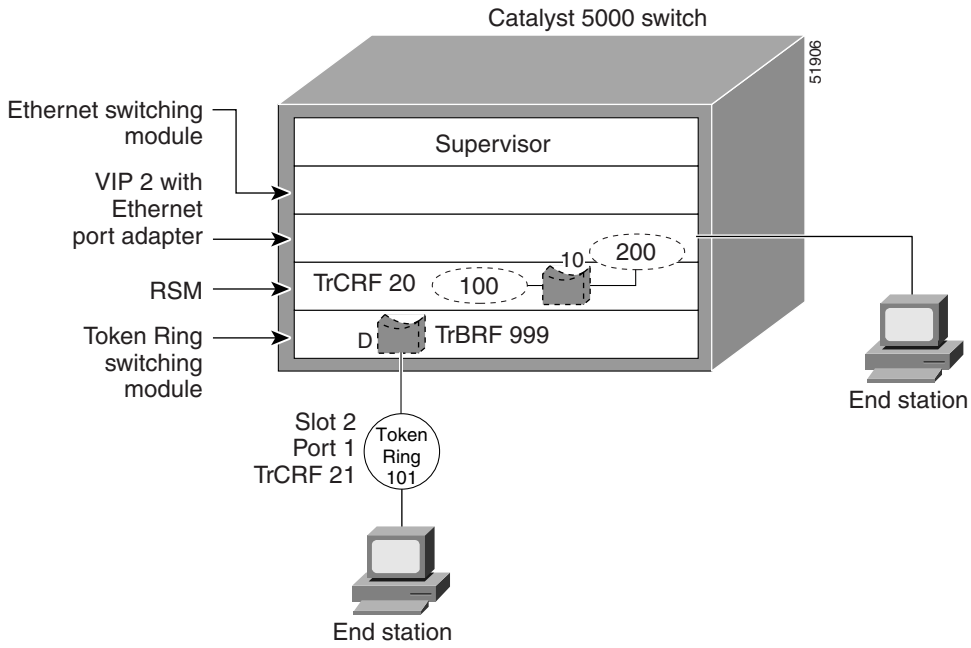
```
set vtp mode server
set vtp v2 enable
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 50 name trbrf50 type trbrf bridge 0xb stp ieee
set vlan 100 name trcrf100 type trcrf ring 0x64 parent 50 mode srt
#add token port to trcrf 100
set vlan 100 5/14
```

SR/TLB Between a TrBRF VLAN and a VIP2 Ethernet Interface Example

The following is the configuration for the RSM as shown in Figure 82:

```
source-bridge ring-group 100
bridge 1 protocol ieee
source-bridge transparent 100 200 10 1
!
interface vlan999 type trbrf
  source-bridge trcrf-vlan 20 ring-group 100
  source-bridge spanning
!
interface ethernet1/0
  bridge-group 1
```

Figure 82 SR/TLB Between a TrBRF VLAN and a VIP2 Ethernet Interface



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 2 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port on 2/1 is assigned to VLAN 21.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
```

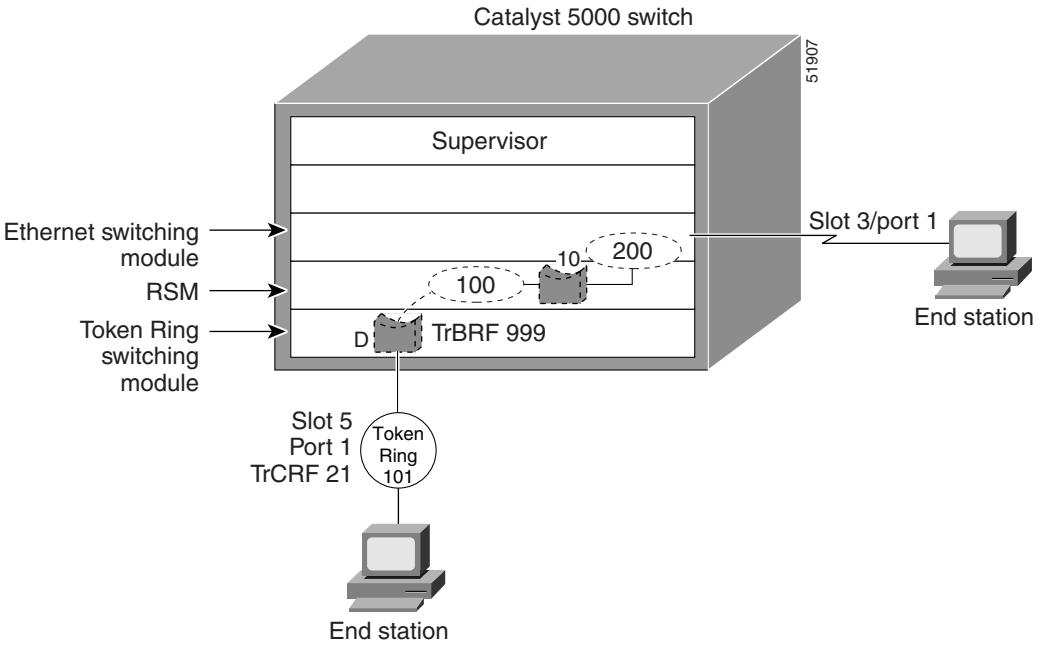
```
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 type trbrf bridge 0xd stp ibm
set vlan 20 type trcrf parent 999 ring 0x64 mode srb
set vlan 21 type trcrf parent 999 ring 0x65 mode srb
#add token ring port to trcrf 21
set vlan 21 2/1
```

SR/TLB Between a TrBRF VLAN and Ethernet VLAN Example

The following is the configuration for the RSM as shown in Figure 83:

```
source-bridge ring-group 100
bridge 1 protocol ieee
source-bridge transparent 100 200 10 1
!
interface vlan999 type trbrf
 source-bridge trcrf-vlan 20 ring-group 100
 source-bridge spanning
!
interface vlan100 type ethernet
 bridge-group 1
```

Figure 83 SR/TLB Between a TrBRF VLAN and Ethernet VLAN



The following is the configuration for the Catalyst 5000 switch with an Ethernet module in slot 3 and a Token Ring switch module in slot 5. In this configuration, the Token Ring port on 5/1 is assigned to TrCRF 21, and the Ethernet port on 3/1 is assigned to VLAN 100.

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
#drip
```

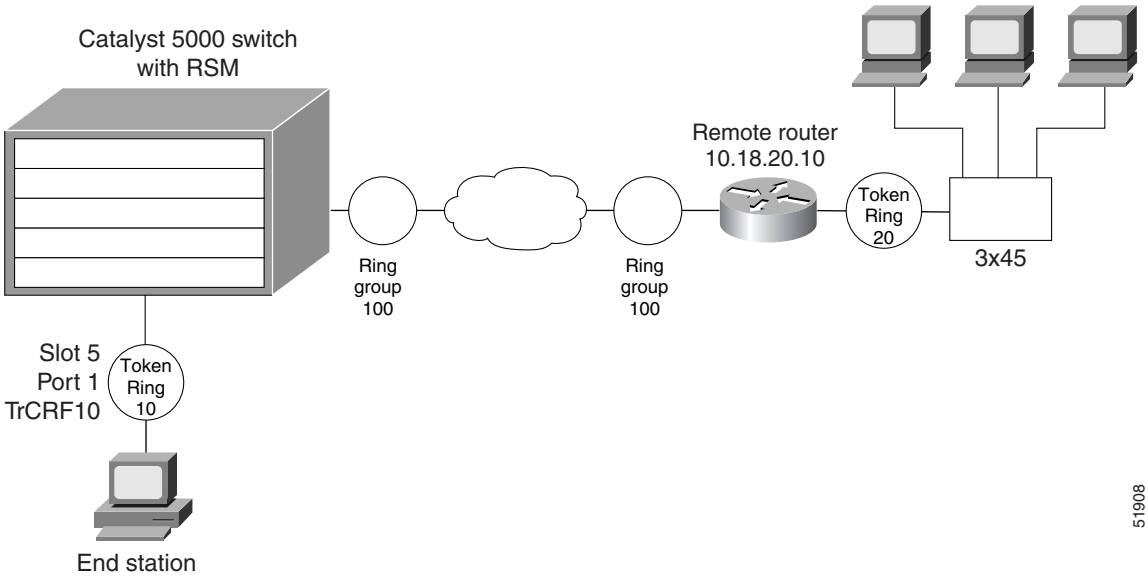
```
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 type trbrf bridge 0xd stp ibm
set vlan 20 type trcrf parent 999 ring 0x64 mode srb
set vlan 21 type trcrf parent 999 ring 0x65 mode srb
#add token port to trcrf 21
set vlan 21 5/1
#add ethernet
set vlan 100 type ethernet
set vlan 100 3/1
```

DLSw+ Example

The following is the configuration for the RSM with a VIP2 serial interface as shown in Figure 84:

```
source-bridge ring-group 100
dlsw local-peer peer-id 10.18.20.8
dlsw remote-peer 0 tcp 10.18.20.6
!
interface Serial1/0
 ip address 10.18.20.8 255.255.255.0
 no keepalive
 clockrate 64000
!
interface Vlan89 type trbrf
 no ip address
 source-bridge trcrf-vlan 9 ring-group 100
 source-bridge spanning
```

Figure 84 Configuration for DLSw+



The following is the configuration for a Catalyst 5000 with a Token Ring module in slot 5. In this configuration, the Token Ring port 5/1 is assigned with the TrCRF VLAN 10:

```
#vtp
set vtp domain trrsm
set vtp mode server
set vtp v2 enable
```

```
#drip
set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 89 type trbrf bridge 0xd stp ibm
set vlan 10 type trcrf parent 89 ring 0xa mode srb
#add token ring port to TrCRF 10
set vlan 10 5/1
set vlan 9 type trcrf parent 89 ring 0x64 mode srb
```

The following is the configuration for the remote router:

```
source-bridge ring group 100
dlsw local-peer peer-id 10.18.20.6
dlsw remote-peer 0 tcp 10.18.20.8
!
interface loopback 0
 ip address 10.18.20.6
!
interface serial1/0
 no ip address
 no keepalive
 clockrate 64000
!
interface tokenring 2/0
 no ip address
 ring-speed 16
 source-bridge 20 1 100
 source-bridge spanning
```

IBM Networking

Overview of IBM Networking

The IBM networking technologies described in this publication can be categorized as network-related or host-related technologies. The IBM Networking section of the *Cisco IOS Bridging and IBM Networking Configuration Guide* discusses the following network-related software components:

The IBM Networking section of the *Cisco IOS Bridging and IBM Networking Configuration Guide* discusses the following host-related software and hardware components:

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

This overview chapter gives a high-level description of each technology. For configuration information, refer to the corresponding chapters in this publication.

Note All commands supported on the Cisco 7500 series routers are also supported on the Cisco 7000 series routers.

RSRB

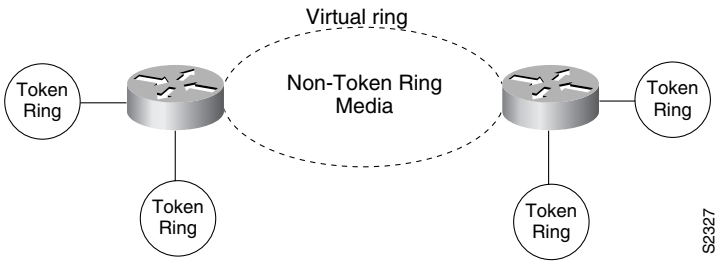
In contrast to Source-Route Bridging (SRB), which involves bridging between Token Ring media only, RSRB is a Cisco technique for connecting Token Ring networks over *non-Token Ring* network segments. (DLSw+ is the Cisco strategic method for providing this function.)

The Cisco RSRB software implementation includes the following features:

- Provides for multiple routers separated by non-Token Ring segments. Three options are available:
  - Encapsulate the Token Ring traffic inside IP datagrams passed over a Transmission Control Protocol (TCP) connection between two routers.
  - Use Fast-Sequenced Transport (FST) to transport RSRB packets to their peers without TCP or User Datagram Protocol (UDP) header or processor overhead.
  - Use data link layer encapsulations over a single serial line, Ethernet, Token Ring, or Fiber Distributed Data Interface (FDDI) ring connected between two routers attached to Token Ring networks.
- Provides for configurable limits to the size of the TCP backup queue.

Figure 85 shows an RSRB topology. The virtual ring can extend across any non-Token Ring media supported by RSRB, such as serial, Ethernet, FDDI, and WANs. The type of media you select determines the way you set up RSRB.

Figure 85 RSRB Topology



Note If you bridge across Token Ring media, it is recommended that you do not use RSRB. Use SRB instead. Refer to the chapter "Configuring Source-Route Bridging" for more information.

Configuration Considerations

Use IP encapsulation only over a TCP connection within complex meshed networks to support connections between peers that are separated by multiple hops and can potentially use multiple paths, and where performance is not an issue. Use direct encapsulation in point-to-point connections. In a point-to-point configuration, using TCP adds unnecessary processing overhead. Multiple peer types, however, can be combined to in a single router by following the directions for each peer type. For example, for a peer to support both TCP and FST remote-peers, you would need to define both a **source-bridge fst** peername and a **source-bridge remote-peer** command for the local router, using the same local IP address.

FST is fast-switched when it receives or sends frames from Ethernet, Token Ring, or FDDI interfaces. It is also fast-switched when it sends and receives from serial interfaces configured with the High-Level Data Link Control (HDLC) encapsulation. In all other cases, FST is slow-switched.

In cases where FST is fast-switched, in either the Cisco routers configured for FST or in the routers contained within the IP “cloud” between a pair of FST peers, only one path is used at a given time between the two FST peers. A single path greatly decreases the likelihood that frames arrive out of sequence. In the rare cases where frames do arrive out of sequence, the FST code on the receiving peer discards the out-of-order frame. Thus the Token Ring end hosts rarely lose a frame over the FST router cloud, and performance levels remain adequate.

The same conditions are true for any slow-switched topology that provides only a single path (for example, a single X.25 network cloud) between the peers. Similarly, if two slow-switched paths are of very different costs such that one always will be chosen over the other, the chances of having frames received out of sequence are also rare.

However, if two or more slow-switched paths of equal cost exist between the two routers (such as two parallel X.25 networks), the routers alternate in sending packets between the two or more equal-cost paths. This results in a high probability of frames arriving out of sequence at the receiver. In such cases, the FST code disposes of every out-of-sequence packet, leading to a large number of drops. This requires that the end hosts resend frames, greatly reducing overall throughput.

When parallel paths exist, we strongly recommend choosing one as the preferred path. Choose a preferred path by specifying a higher bandwidth for the path that contains the direct connections to the two or more parallel paths on the router.

Do not use FST when the probability exists for frames to lose their order in your network. If you have a network where frames are routinely reordered, it is better to use the TCP protocol for RSRB. TCP provides the overhead necessary to bring frames back in order on the receiving router. FST, to remain fast, does not provide for such a mechanism, and will discard out-of-order frames.

Logical Link Control, type 2 (LLC2) local acknowledgment can be enabled only with TCP remote peers (as opposed to LAN or direct serial interface remote peers) because the Cisco IOS software needs the reliability of TCP to provide the same reliability that an LLC2 LAN end-to-end connection provides. Therefore, the direct media encapsulation options for the **source-bridge remote-peer** command cannot be used.

If the LLC2 session between the local host and the router terminates on either side of the connection, the other device will be informed to terminate its connection to its local host.

If the TCP queue length of the connection between the two routers reaches 90 percent of its limit, they send Receiver-not-Ready (RNR) messages to the local hosts until the queue limit is reduced to below this limit.

The configuration of the LLC2 parameters for the local Token Ring interfaces can affect overall performance. Refer to the “Configuring LLC2 and SDLC Parameters” chapter for more details about fine-tuning your network through the LLC2 parameters.

Note As previously stated, local acknowledgment for LLC2 is meant only for extreme cases in which communication is not possible otherwise. Because the router must maintain a full LLC2 session, the number of simultaneous sessions it can support before performance degrades depends on the mix of other protocols and their loads.

The routers at each end of the LLC2 session execute the full LLC2 protocol, which can result in some overhead. The decision to turn on local acknowledgment for LLC2 should be based on the speed of the backbone network in relation to the Token Ring speed. For LAN segments separated by slow-speed serial links (for example, 56 kbps), the T1 timer problem could occur more frequently. In such cases, it might be wise to turn on local acknowledgment for LLC2. For LAN segments separated by a FDDI backbone, backbone delays will be minimal; in such cases, local acknowledgment for LLC2 should not be turned on. Speed mismatch between the LAN segments and the backbone network is one criterion to be used in the decision to use local acknowledgment for LLC2.

There are some situations (such as host B failing between the time host A sends data and the time host B receives it) in which host A would behave as if, *at the LLC2 layer*, data was received when it actually was not, because the device acknowledges that it received data from host A before it confirms that host B can actually receive the data. But because both NetBIOS and SNA have error recovery in situations where an end device goes down, these higher-level protocols will resend any missing or lost data. These transaction request/confirmation protocols exist above LLC2, so they are not affected by tight timers, as is LLC2. They also are transparent to local acknowledgment.

If you are using NetBIOS applications, note that there are two NetBIOS timers—one at the link level and one at the next higher level. Local acknowledgment for LLC2 is designed to solve session timeouts at the link level only. If you are experiencing NetBIOS session timeouts, you have two options:

- Experiment with increasing your NetBIOS timers.
- Avoid using NetBIOS applications on slow serial lines.

In a configuration scenario where RSRB is configured between Router A and Router B and both routers are not routing IP, a Host connected to router A through Token Ring (or other LAN media) has no IP connectivity to router B. This restriction exists because IP datagrams received from the Host by Router A are encapsulated and sent to router B where they can only be de-encapsulated and source-bridged to a Token Ring. In this scenario, IP routing is recommended. To enable the Host to reach Router B in this scenario, IP routing should be enabled on Router A's Token Ring interface to which the Host is attached.

DLSw+

Data-Link Switching Plus (DLSw+) is a method of transporting SNA and NetBIOS. It complies with the DLSw standard documented in RFC 1795 and the DLSw Version 2 standard. DLSw+ is an alternative to RSRB that addresses several inherent problems that exist in RSRB, such as:

- SRB hop-count limits (SRB's limit is seven)
- Broadcast traffic (including SRB explorer frames or NetBIOS name queries)
- Unnecessary traffic (acknowledgments and keepalives)
- Data-link control timeouts

This section contains a brief overview of DLSw+:

DLSw Standard

The DLSw standard, documented in RFC 1795, defines the switch-to-switch protocol between DLSw routers. The standard also defines a mechanism to terminate data-link control connections locally and multiplex the traffic from the data-link control connections to a TCP connection. The standard always calls for the transport protocol to be TCP and always requires that data-link control connections be locally terminated (the equivalent of the Cisco local acknowledgment option). The standard also requires that the SRB RIF be terminated at the DLSw router. The standard describes a means for prioritization and flow control and defines error recovery procedures that ensure data-link control connections are appropriately disabled if any part of their associated circuits breaks.

The DLSw standard does not specify when to establish TCP connections. The capabilities exchange allows compliance to the standard, but at different levels of support. The standard does not specify how to cache learned information about MAC addresses, RIFs, or NetBIOS names. It also does not describe how to track either capable or preferred DLSw partners for either backup or load-balancing purposes.The standard does not provide the specifics of media conversion, but leaves the details up to the implementation. It does not define how to map switch congestion to the flow control for data-link control. Finally, the MIB is documented under a separate RFC.

DLSw Version 2 Standard

In the Version 1 standard, a network design requires fully meshed connectivity so that all peers were connect to every other peer. This design creates unnecessary broadcast traffic because an explorer propagates to every peer for every broadcast.

The Version 2 standard is documented in RFC 2166. It includes RFC 1795 and adds the following enhancements:

Users implement DLSw+ Version 2 for scalability if they are using multivendor DLSw devices with an IP multicast network. DLSw Version 2 requires complex planning because it involves configuration changes across an IP network.

IP Multicast

Multicast service avoids duplication and excessive bandwidth of broadcast traffic because it replicates and propagates messages to its multicast members only as necessary. It reduces the amount of network overhead in the following ways:

- Avoids the need to maintain TCP Switch-to-Switch Protocol (SSP) connections between two DLSw peers when no circuits are available
- Ensures that each broadcast results in only a single explorer over every link

DLSw Version 2 is for customers who run a multicast IP network and do not need the advantages of border peering.

UDP Unicast

DLSw Version 2 uses UDP unicast in response to a IP multicast. When address resolution packets (CANUREACH_EX, NETBIOS_NQ_ex, NETBIOS_ANQ, and DATAFRAME) are sent to multiple destinations (IP multicast service) DLSw Version 2 sends the response frames (ICANREACH_ex and NAME_RECOGNIZED_ex) via UDP unicast.

Enhanced Peer-on-Demand Routing Feature

DLSw Version 2 establishes TCP connections only when necessary and the TCP connections are brought down when there are no circuits to a DLSw peer for a specified amount of time. This method, known as peer-on-demand routing, was recently introduced in DLSw Version 2, but has been implemented in Cisco DLSw+ border peer technology since Cisco IOS Release 10.3.

Expedited TCP Connection

DLSw Version 2 efficiently establishes TCP connections. Previously, DLSw created two unidirectional TCP connections and then disconnected one after the capabilities exchange took place. With DLSw Version 2, a single bidirectional TCP connection establishes if the peer is brought up as a result of an IP multicast/UDP unicast information exchange.

DLSw+ Features

DLSw+ is the Cisco version of DLSw and it supports several additional features and enhancements. DLSw+ is a means of transporting SNA and NetBIOS traffic over a campus or WAN. The end systems can attach to the network over Token Ring, Ethernet, Synchronous Data Link Control (SDLC) Protocol, Qualified Logical Link Control (QLLC), or FDDI. See the *DLSw+ Design and Implementation Guide* Appendix B, “DLSw+ Support Matrix,” for details. DLSw+ switches between diverse media and locally terminates the data links, keeping acknowledgments, keepalives, and polling off the WAN. Local termination of data links also eliminates data-link control timeouts that can occur during transient network congestion or when rerouting around failed links. Finally, DLSw+ provides a mechanism for dynamically searching a network for SNA or NetBIOS resources and includes caching algorithms that minimize broadcast traffic.

This section contains information on the following topics related to DLSw+ features:

DLSw+ is fully compatible with any vendor's RFC 1795 implementation and the following features are available when both peers are using DLSw+:

- Peer groups and border peers
- Backup peers
- Promiscuous and on-demand peers
- Explorer firewalls and location learning
- NetBIOS dial-on-demand routing feature support
- UDP unicast support
- Load balancing
- Support for LLC1 circuits
- Support for multiple bridge groups
- Support for RIF Passthru
- SNA type of service feature support
- Local acknowledgment for Ethernet-attached devices and media conversion for SNA PU 2.1 and PU 2.0 devices
- Conversion between LLC2 to SDLC between PU 4 devices
- Local or remote media conversion between LANs and either the SDLC Protocol or QLLC
- SNA View, Blue Maps, and Internetwork Status Monitor (ISM) support

MIB enhancements that allow DLSw+ features to be managed by the CiscoWorks Blue products, SNA Maps, and SNA View. Also, new traps alert network management stations of peer or circuit failures. For more information, refer to the current Cisco IOS release note for the location of the Cisco MIB website.

Local Acknowledgment

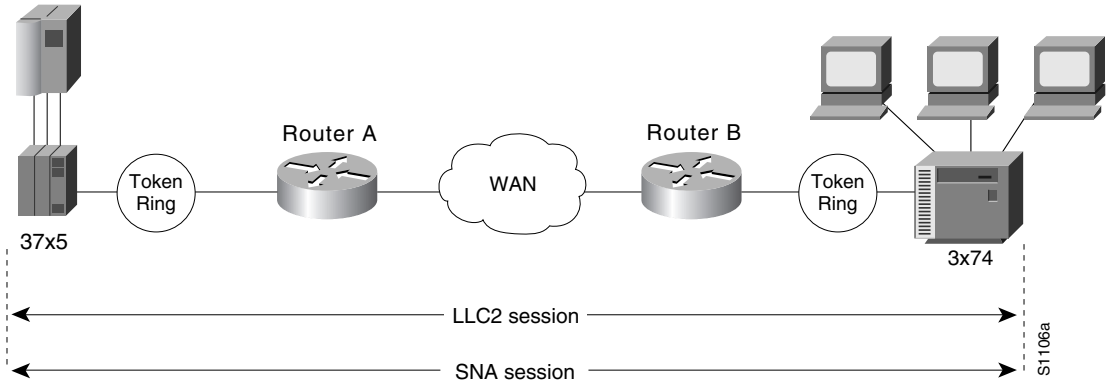
When you have LANs separated by wide geographic distances, and you want to avoid multiple resending or loss of user sessions that can occur with time delays, encapsulate the source-route bridged traffic inside IP datagrams passed over a TCP connection between two routers with local acknowledgment enabled.

LLC2 is an ISO standard data-link level protocol used in Token Ring networks. LLC2 was designed to provide reliable sending of data across LAN media and to cause minimal or at least predictable time delays. However, RSRB and WAN backbones created LANs that are separated by wide, geographic distances-spanning countries and continents. As a result, LANs have time delays that are longer than LLC2 allows for bidirectional communication between hosts. Local acknowledgment addresses the problem of unpredictable time delays, multiple resending, and loss of user sessions.

In a typical LLC2 session, when one host sends a frame to another host, the sending host expects the receiving host to respond positively or negatively in a predefined period of time commonly called the *T1 time*. If the sending host does not receive an acknowledgment of the frame it sent within the T1 time, it retries a few times (normally 8 to 10). If there is still no response, the sending host drops the session.

Figure 86 illustrates an LLC2 session in which a 37x5 on a LAN segment communicates with a 3x74 on a different LAN segment separated via a wide-area backbone network. Frames are transported between Router A and Router B by means of DLSw+. However, the LLC2 session between the 37x5 and the 3x74 is still end-to-end; that is, every frame generated by the 37x5 traverses the backbone network to the 3x74, and the 3x74, on receipt of the frame, acknowledges it.

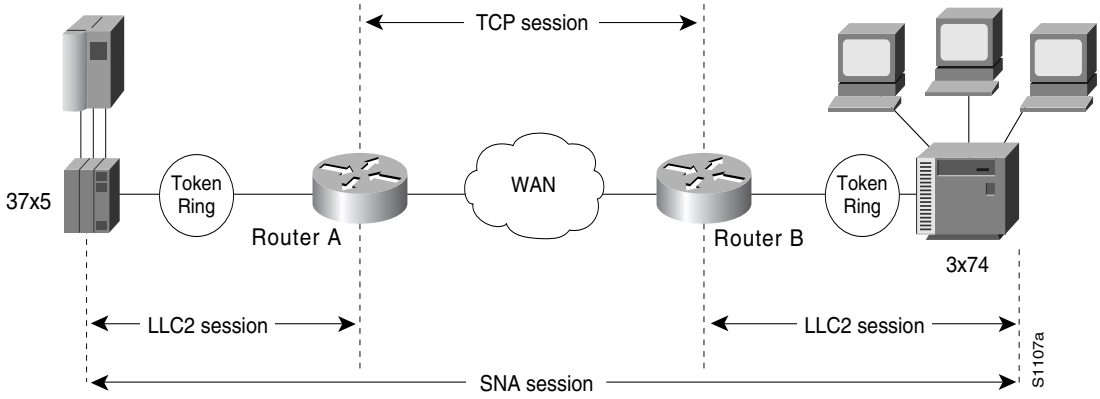
Figure 86 LLC2 Session Without Local Acknowledgment



On backbone networks consisting of slow serial links, the T1 timer on end hosts could expire before the frames reach the remote hosts, causing the end host to resend. Resending results in duplicate frames reaching the remote host at the same time as the first frame reaches the remote host. Such frame duplication breaks the LLC2 protocol, resulting in the loss of sessions between the two IBM machines.

One way to solve this time delay is to increase the timeout value on the end nodes to account for the maximum transit time between the two end machines. However, in networks consisting of hundreds or even thousands of nodes, every machine would need to be reconfigured with new values. With local acknowledgment for LLC2 enabled, the LLC2 session between the two end nodes would not be not end-to-end, but instead, would terminate at two local routers. Figure 87 shows the LLC2 session with the 37x5 ending at Router A and the LLC2 session with the 3x74 ending at Router B. Both Router A and Router B execute the full LLC2 protocol as part of local acknowledgment for LLC2.

Figure 87 LLC2 Session with Local Acknowledgment



With local acknowledgment for LLC2 enabled in both routers, Router A acknowledges frames received from the 37x5. The 37x5 still operates as if the acknowledgments it receives are from the 3x74. Router A looks like the 3x74 to the 37x5. Similarly, Router B acknowledges frames received from the 3x74. The

3x74 operates as if the acknowledgments it receives are from the 37x5. Router B looks like the 3x74 to 37x5. Because the frames do not have to travel the WAN backbone networks to be acknowledged, but are locally acknowledged by routers, the end machines do not time out, resulting in no loss of sessions.

Enabling local acknowledgment for LLC2 has the following advantages:

- Local acknowledgment for LLC2 solves the T1 timer problem without having to change any configuration on the end nodes. The end nodes are unaware that the sessions are locally acknowledged. In networks consisting of hundreds or even thousands of machines, this is a definite advantage. All the frames acknowledged by the Cisco IOS software appear to the end hosts to be coming from the remote IBM machine. In fact, by looking at a trace from a protocol analyzer, one cannot say whether a frame was acknowledged by the local router or by a remote IBM machine. The MAC addresses and the RIFs generated by the Cisco IOS software are identical to those generated by the remote IBM machine. The only way to find out whether a session is locally acknowledged is to use either a **show local-ack** command or a **show source-bridge** command on the router.
- All the supervisory (RR, RNR, REJ) frames that are locally acknowledged go no farther than the router. Without local acknowledgment for LLC2, *every* frame traverses the backbone. With local acknowledgment, only data (I-frames) traverse the backbone, resulting in less traffic on the backbone network. For installations in which customers pay for the amount of traffic passing through the backbone, this could be a definite cost-saving measure. A simple protocol exists between the two *peers* to bring up or down a TCP session.

Notes on Using LLC2 Local Acknowledgment

LLC2 local acknowledgment is enabled with TCP and DLSw+ Lite remote peers.

If the LLC2 session between the local host and the router terminates in either router, the other will be informed to terminate its connection to its local host.

If the TCP queue length of the connection between the two routers reaches the high-water mark, the routers sends Receiver-Not-Ready (RNR) messages to the local hosts until the queue limit is reduced to below this limit. It is possible, however, to prevent the RNR messages from being sent by using the **dlsw llc2 nornr** command.

The configuration of the LLC2 parameters for the local Token Ring interfaces can affect overall performance. Refer to the chapter “Configuring LLC2 and SDLC Parameters” in this manual for more details about fine-tuning your network through the LLC2 parameters.

The routers at each end of the LLC2 session execute the full LLC2 protocol, which could result in some overhead. The decision to use local acknowledgment for LLC2 should be based on the speed of the backbone network in relation to the Token Ring speed. For LAN segments separated by slow-speed serial links (for example, 56 kbps), the T1 timer problem could occur more frequently. In such cases, it might be wise to turn on local acknowledgment for LLC2. For LAN segments separated by a T1, backbone delays will be minimal; in such cases, FST or direct should be considered. Speed mismatch between the LAN segments and the backbone network is one criterion to help you decide to use local acknowledgment for LLC2.

There are some situations (such as the receiving host failing between the time the sending host sends data and the time the receiving host receives it), in which the sending host would determine, *at the LLC2 layer*, that data was received when it actually was not. This error occurs because the router acknowledges that it received data from the sending host before it determines that the receiving host can actually receive the data. But because both NetBIOS and SNA have error recovery in situations where an end device goes down, these higher-level protocols will resend any missing or lost data. Because these transaction request/confirmation protocols exist above LLC2, they are not affected by tight timers, as is LLC2. They also are transparent to local acknowledgment.

If you are using NetBIOS applications, note that there are two NetBIOS timers—one at the link level and one at the next higher level. Local acknowledgment for LLC2 is designed to solve link timeouts only. If you are experiencing NetBIOS session timeouts, you have two options:

- Experiment with increasing your NetBIOS timers and decreasing your maximum NetBIOS frame size.
- Avoid using NetBIOS applications on slow serial lines.

Note By default, the Cisco IOS software translates Token Ring LLC2 to Ethernet 802.3 LLC2. To configure the router to translate Token Ring LLC2 frames into Ethernet 0x80d5 format frames, refer to the section "Enable Token Ring LLC2-to-Ethernet Conversion" in the "Configuring Source-Route Bridging" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

DLSw+ Support for Other SNA Features

DLSw+ can be used as a transport for SNA features such as LAN Network Manager (LNM), DSPU, SNA service point, and SNA Switching Services (SNASw) through a Cisco IOS feature called virtual data-link control (VDLC).

LNM over DLSw+ allows DLSw+ to be used in Token Ring networks that are managed by IBM's LNM software. Using this feature, LNM can be used to manage Token Ring LANs, control access units, and Token Ring attached devices over a DLSw+ network. All management functions continue to operate as they would in a source-route bridged network or an RSRB network.

DSPU over DLSw+ allows the Cisco DSPU feature to operate in conjunction with DLSw+ in the same router. DLSw+ can be used either upstream (toward the mainframe) or downstream (away from the mainframe) of DSPU. DSPU concentration consolidates the appearance of multiple physical units (PUs) into a single PU appearance to VTAM, minimizing memory and cycles in central site resources (VTAM, NCP, and routers) and speeding network startup.

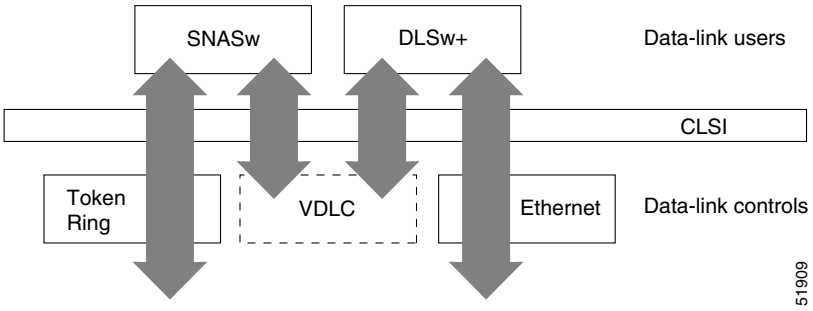
SNA service point over DLSw+ allows the Cisco SNA service point feature to be used in conjunction with DLSw+ in the same router. Using this feature, SNA service point can be configured in remote routers, and DLSw+ can provide the path for the remote service point PU to communicate with NetView. This allows full management visibility of resources from a NetView 390 console, while concurrently offering the value-added features of DLSw+ in an SNA network.

SNASw over DLSw+ allows the Cisco APPN Branch Extender functionality to be used in conjunction with DLSw+ in the same router. With this feature, DLSw+ can be used to access SNASw in the data center. DLSw+ can also be used as a transport SNASw upstream connectivity, providing nondisruptive recovery from failures. The DLSw+ network can appear as a connection network to the SNASw nodes.

Using DLSw+ as a transport for other Cisco IOS SNA features requires a feature called VDLC. Cisco IOS data-link users (such as LNM, DSPU, SNA service point, and SNASw) write to a virtual data-link control interface. DLSw+ then reads from this interface and sends out the traffic. Similarly, DLSw+ can receive traffic destined for one of these Data Link Users and write it to the virtual data-link control interface, from which the appropriate Data Link User will read it.

In Figure 88, SNASw and DLSw+ use Token Ring and Ethernet, respectively, as "real" data-link controls, and use virtual data-link control to communicate between themselves. When one of the high-layer protocols passes data to the virtual data-link control, the virtual data-link control must pass it to a higher-layer protocol; nothing leaves the virtual data-link control without going through a data-link user.

Figure 88 VDLC Interaction with Higher-Layer Protocols



The higher-layer protocols make no distinction between the VDLC and any other data-link control, but they do identify the VDLC as a destination. In the example shown in , SNASw has two ports: a physical port for Token Ring and a logical (virtual) port for the VDLC. In the case of the SNASw VDLC port, when you define the SNASw VDLC port, you can also specify the MAC address assigned to it. That means data going from DLSw+ to SNASw by way of the VDLC is directed to the VDLC MAC address. The type of higher-layer protocol you use determines how the VDLC MAC address is assigned.

STUN and BSTUN

The Cisco IOS software supports serial tunnel (STUN) and block serial tunnel (BSTUN). Our BSTUN implementation enhances Cisco 2500, 4000, 4500, 4700, 7200 series routers to support devices that use the Binary Synchronous Communication (Bisync) data-link protocol and asynchronous security protocols that include Adplex, ADT Security Systems, Inc., Diebold, and asynchronous generic traffic. BSTUN implementation is also supported on the 4T network interface module (NIM) on the Cisco 4000 and 4500 series routers. Our support of the bisync protocol enables enterprises to transport Bisync traffic and SNA multiprotocol traffic over the same network.

This section contains the following topics:

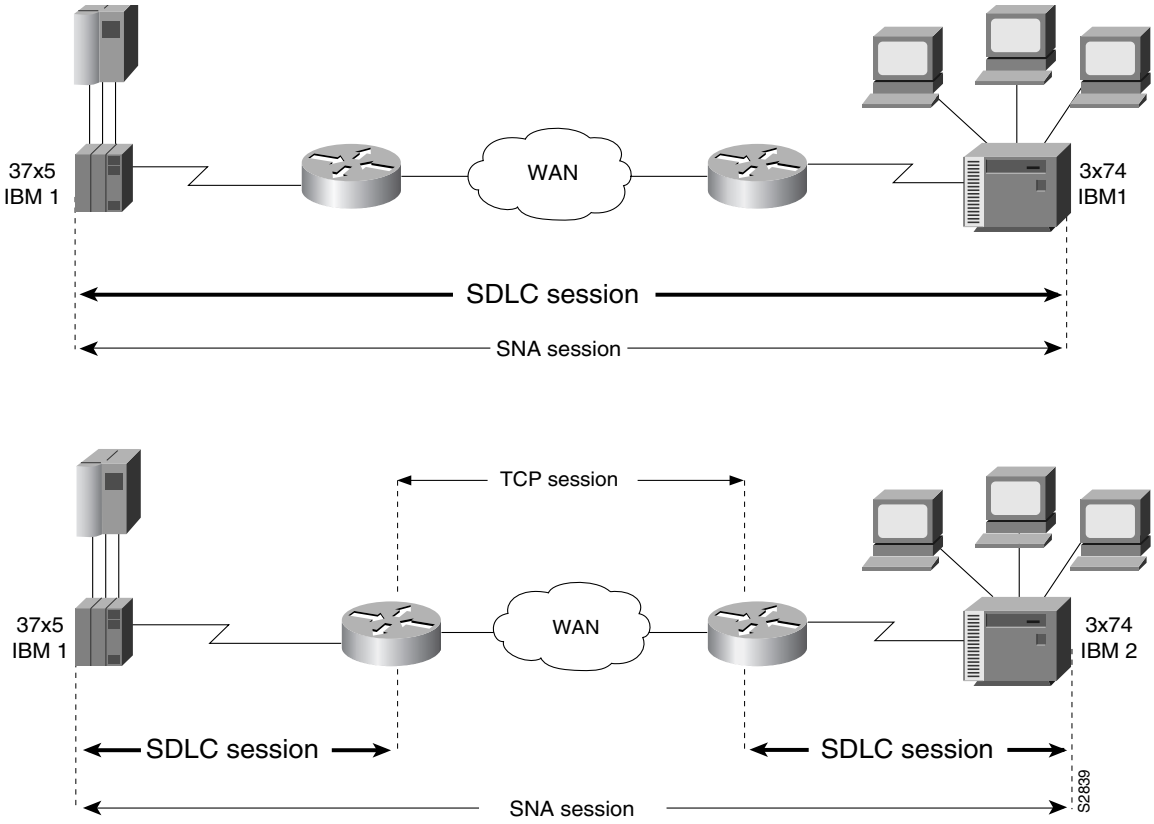
STUN Networks

STUN operates in two modes: passthrough and local acknowledgment. Figure 89 shows the difference between passthrough mode and local acknowledgment mode.

The upper half of Figure 89 shows STUN configured in passthrough mode. In passthrough mode, the routers act as a wire and the SDLC session remains between the end stations. In this mode, STUN provides a straight passthrough of all SDLC traffic, including control frames.

The lower half of Figure 89 shows STUN configured in local acknowledgment mode. In local acknowledgment mode, the routers terminate the SDLC sessions and send only data across the WAN. Control frames no longer travel the WAN backbone networks.

Figure 89 Comparison of STUN in Passthrough Mode and Local Acknowledgment Mode



Note To enable STUN local acknowledgment, you first enable the routers for STUN and configure them to appear on the network as primary or secondary SDLC nodes. TCP/IP encapsulation must be enabled. The Cisco STUN local acknowledgment feature also provides priority queueing for TCP-encapsulated frames.

STUN Features

The Cisco STUN implementation provides the following features:

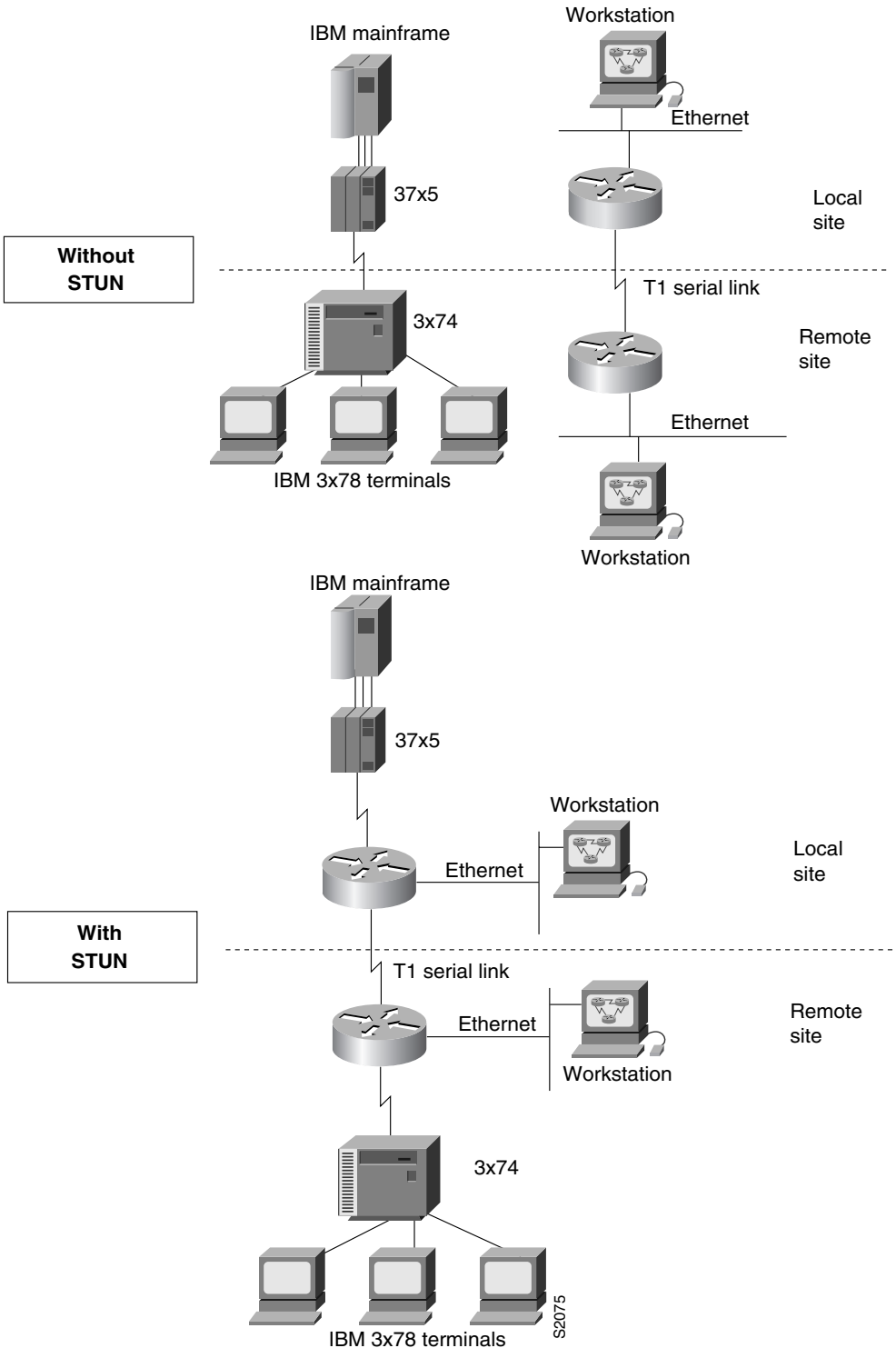
- Encapsulates SDLC frames in either the Transmission Control Protocol/Internet Protocol (TCP/IP) or the HDLC protocol.
- Allows two devices using SDLC- or HDLC-compliant protocols that are normally connected by a direct serial link to be connected through one or more Cisco routers, reducing leased-line costs.

  When you replace direct serial links with routers, serial frames can be propagated over arbitrary media and topologies to another router with a STUN link to an appropriate endpoint. The intervening network is not restricted to STUN traffic, but rather, is multiprotocol. For example, instead of running parallel backbones for DECnet and SNA/SDLC traffic, this traffic now can be integrated into an enterprise backbone network.
- Supports local acknowledgment for direct Frame Relay connectivity between routers, without requiring TCP/IP.

- Allows networks with IBM mainframes and communications controllers to share data using Cisco routers and existing network links. As an SDLC function, STUN fully supports the IBM SNA and allows IBM SDLC frames to be sent across the network media and shared serial links. illustrates a typical network configuration without STUN and the same network configured with STUN.
- Encapsulates SDLC frame traffic packets and routes them over any of the supported network media (serial, FDDI, Ethernet, and Token Ring, X.25, SMDS, and T1/T3) using TCP/IP encapsulation. Because TCP/IP encapsulation is used, you can use any of the Cisco routing protocols to route the packets.
- Copies frames to destinations based on address. STUN in passthrough mode does not modify the frames in any way or participate in SDLC windowing or resending; these functions are left to the communicating hosts. However, STUN in local acknowledgment mode does participate in SDLC windowing and resending through local termination of the SDLC session.
- Ensures reliable sending of data across serial media having minimal or predictable time delays. With the advent of STUN and WAN backbones, serial links now can be separated by wide geographic distances spanning countries and continents. As a result, these serial links have time delays that are longer than SDLC allows for bidirectional communication between hosts. The STUN local acknowledgment feature addresses the problems of unpredictable time delays, multiple resending, or loss of sessions.
- Allows for configuration of redundant links to provide transport paths if part of the network goes down.

Figure 90 shows the difference between an IBM network with STUN and one without STUN.

Figure 90 IBM Network Configuration without STUN and with STUN



BSTUN Networks

The Bisync feature enables your Cisco 2500, 3600, 4000, 4500, 4700, and 7200 series router to support devices that use the Bisync data-link protocol. This protocol enables enterprises to transport Bisync traffic over the same network that supports their SNA and multiprotocol traffic, eliminating the need for separate Bisync facilities.

At the access router, traffic from the attached Bisync device is encapsulated in IP. The Bisync traffic can then be routed across arbitrary media to the host site where another router supporting Bisync will remove the IP encapsulation headers and present the Bisync traffic to the Bisync host or controller over a serial connection. HDLC can be used as an alternative encapsulation method for point-to-point links.

BSTUN Features

The Cisco implementation of BSTUN provides the following features:

- Encapsulates Bisync, Adplex, ADT Security Systems, Inc., Diebold, asynchronous generic, and Monitor Dynamics Inc., traffic for transfer over router links. The tunneling of asynchronous security protocols (ASP) feature enables your Cisco 2500, 3600, 4000, 4500, or 7200 series router to support devices that use the following asynchronous security protocols:
  - adplex
  - adt-poll-select
  - adt-vari-poll
  - diebold
  - async-generic
  - mdi
- Provides a tunnel mechanism for BSTUN over Frame Relay, without using TCP/IP encapsulation.
- Supports Bisync devices and host applications without modification.
- Uses standard synchronous serial interfaces on Cisco 2500 series and the 4T network interface module (NIM) on the Cisco 4000 series and Cisco 4500 series.
- Supports point-to-point, multidrop, and virtual multidrop configurations.

Note The async-generic item is not a protocol name. It is a command keyword used to indicate generic support of other asynchronous security protocols that are not explicitly supported.

LLC2 and SDLC Parameters

The LLC2 and SDLC protocols provide data link layer support for higher-layer network protocols and features such as SDLC Logical Link Control (SDLLC) and RSRB with local acknowledgment. The features that are affected by LLC2 parameter settings are listed in the "The Cisco Implementation of LLC2" section on page 16. The features that require SDLC configuration and use SDLC parameters are listed in the "The Cisco Implementation of SDLC" section on page 17.

LLC2 and SDLC package data in frames. LLC2 and SDLC stations require acknowledgments from receiving stations after a set amount of frames have been sent before sending further data. The tasks described in this chapter modify default settings regarding the control field of the data frames. By

modifying the control field parameters, you can determine the number of acknowledgments sent for frames received and the level of polling used to determine available stations. In this manner, you can set the amount of resources used for frame checking and optimize the network load.

SDLC is used as the primary SNA link-layer protocol for WAN links. SDLC defines two types of network nodes: primary and secondary. Primary nodes poll secondary nodes in a predetermined order. Secondary nodes then send any outgoing data. When configured as primary and secondary nodes, our routers are established as SDLC stations.

The Cisco Implementation of LLC2

The Cisco LLC2 implementation supports the following features:

- Local acknowledgment for RSRB

  This feature is used in our implementation of RSRB as described in the chapter "Configuring Source-Route Bridging."

  Because LANs are now connected through RSRB and WAN backbones, the delays that occur are longer than LLC2 allows for bidirectional communication between hosts. Our local acknowledgment feature addresses the problem of delays, resending, and loss of user sessions.

- IBM LNM support

  Routers using 4- or 16-Mbps Token Ring interfaces configured for SRB support Lan Network Manager (LNM) and provide all IBM bridge program functions. With LNM, a router appears as an IBM source-route bridge, and can manage or monitor any connected Token Ring interface.

  LNM support is described in the chapter "Configuring Source-Route Bridging."

- SDLLC media translation

  The SDLLC feature provides media translation between the serial lines running SDLC and Token Rings running LLC2. SDLLC consolidates the IBM SNA networks running SDLC into a LAN-based, multiprotocol, multimedia backbone network.

  SDLLC is described in the chapter "Configuring IBM Network Media Translation."

- ISO Connection-Mode Network Service (CMNS)

  The Cisco CMNS implementation runs X.25 packets over LLC2 so that X.25 can be extended to Ethernet, FDDI, and Token Ring media.

The Cisco Implementation of SDLC

The Cisco SDLC implementation supports the following features:

- Frame Relay Access Support (FRAS)

  With FRAS, a router functions as a Frame Relay Access Device (FRAD) for SDLC, Token Ring, and Ethernet-attached devices over a Frame Relay Boundary Network Node (BNN) link.

  Frame Relay access support is described in the chapter "Configuring SNA Frame Relay Access Support."

- SDLLC media translation

  The SDLLC feature provides media translation between the serial lines running SDLC and Token Rings running LLC2. SDLLC consolidates the IBM SNA networks running SDLC into a LAN-based, multiprotocol, multimedia backbone network.

  SDLLC is described in the chapter "Configuring IBM Network Media Translation."

- SDLC local acknowledgment

  SDLC local acknowledgment is used with SDLC STUN. TCP/IP must be enabled. With local acknowledgment, STUN SDLC connections can be terminated locally at the router, eliminating the need for acknowledgments to be sent across a WAN.

  SDLC local acknowledgment is described in the section "Establish the Frame Encapsulation Method" in the chapter "Configuring STUN and BSTUN."

IBM Network Media Translation

The Cisco IOS software includes the following media translation features that enable network communications across heterogeneous media:

- SDLLC media translation enables a device on a Token Ring to communicate with a device on a serial link.
- QLLC conversion enables an IBM device to communicate with an X.25 network without having to install the X.25 software on local IBM equipment.

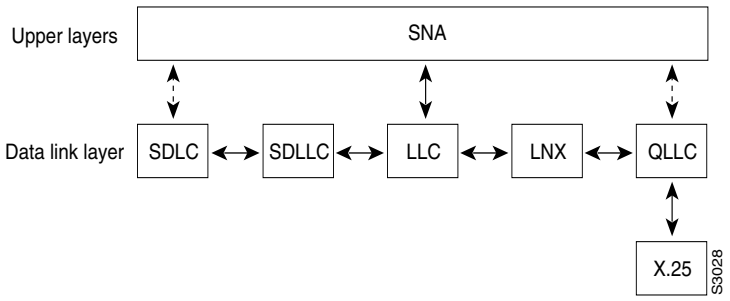
SDLLC is a Cisco Systems proprietary software feature that enables a device on a Token Ring to communicate with a device on a serial link by translating between LLC2 and SDLC at the link layer.

SNA uses SDLC and LLC2 as link layer protocols to provide a reliable connection. The translation function between these industry-standard protocols takes place in the proprietary Cisco software.

This section contains a brief overview of IBM Network Media Translation:

Figure 91 illustrates how SDLLC provides data link layer support for SNA communication.

Figure 91 SNA Data Link Layer Support



SDLLC Media Translation Features

The SDLLC feature allows a PU 4, PU 2.1, or PU 2 to communicate with a PU 2 SDLC device as follows:

- SDLLC with direct connection—A 37x5 front-end processor (FEP) on a Token Ring and the 3x74 cluster controller connected to a serial line are each connected to an interface on the same router configured with SDLLC.
- SDLLC with RSRB—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB using direct encapsulation, RSRB over an FST connection, or RSRB over a TCP connection.
- SDLLC with RSRB and local acknowledgment—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB over a TCP connection that has local acknowledgment enabled.

In all these topologies, each IBM end node (the FEP and cluster controller) has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were communicating over a serial line. That is, the SDLLC software makes translation between the two media transparent to the end nodes.

Virtual Token Ring Concept

Central to the Cisco SDLLC feature is the concept of a virtual Token Ring device residing on a virtual Token Ring. Because the Token Ring device expects the node with which it is communicating also to be on a Token Ring, each SDLLC device on a serial line must be assigned an SDLLC virtual Token Ring address (SDLLC VTRA). Like real Token Ring addresses, SDLLC VTRAs must be unique across the network.

In addition to the SDLLC VTRA, an SDLLC virtual ring number must be assigned to each SDLLC device on a serial line. (The SDLLC virtual ring number differs from the virtual ring group numbers that are used to configure RSRB and multiport bridging.)

As part of its virtual telecommunications access method (VTAM) configuration, the IBM node on the Token Ring has knowledge of the SDLLC VTRA of the serial device with which it communicates. The SDLC VTRA and the SDLLC virtual ring number are a part of the SDLLC configuration for the router's serial interface. When the Token Ring host sends out explorer packets with the SDLLC VTRA as the destination address in the MAC headers, the router configured with that SDLLC VTRA intercepts the frame, fills in the SDLLC virtual ring number address and the bridge number in the RIF, then sends the response back to the Token Ring host. A route is then established between the Token Ring host and the router. After the Cisco IOS software performs the appropriate frame conversion, the system uses this route to forward frames to the serial device.

Resolving Differences in LLC2 and SDLC Frame Size

IBM nodes on Token Ring media normally use frame sizes greater than 1 KB, whereas the IBM nodes on serial lines normally limit frame sizes to 265 or 521 bytes. To reduce traffic on backbone networks and provide better performance, Token Ring nodes should send frames that are as large as possible. As part of the SDLLC configuration on the serial interface, the largest frame size the two media can support should be selected. The Cisco IOS software can fragment the frames it receives from the Token Ring device before forwarding them to the SDLC device, but it does not assemble the frames it receives from the serial device before forwarding them to the Token Ring device.

Maintaining a Dynamic RIF Cache

SDLLC maintains a dynamic RIF cache and caches the entire RIF; that is, the RIF from the source station to destination station. The cached entry is based on the best path at the time the session begins. SDLLC uses the RIF cache to maintain the LLC2 session between the router and the host FEP. SDLLC does not age these RIF entries. Instead, SDLLC places an entry in the RIF cache for a session when the session begins and flushes the cache when the session terminates. You cannot flush these RIFs because if you flush the RIF entries randomly, the Cisco IOS software cannot maintain the LLC2 session to the host FEP.

Other Considerations

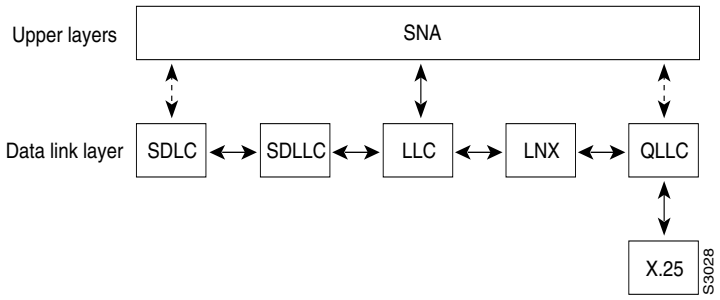
The following are additional facts regarding SDLC and SDLLC:

- As part of the Cisco SDLC implementation, only modulus 8 Normal Response Mode (NRM) sessions are maintained for the SDLC session.
- SDLC sessions are always locally acknowledged. LLC2 sessions can be optionally configured for local acknowledgment.
- SDLLC does not apply to SNA subarea networks, such as 37x5 FEP-to-37x5 FEP communication.
- Parameters such as the maximum number of information frames (I-frames) outstanding before acknowledgment, frequency of polls, and response time to poll frames can be modified per interface. If local acknowledgment is not enabled, these parameters are modified on the SDLC interface. If local acknowledgment is enabled, these parameters are modified on the Token Ring interface.
- Local acknowledgment only applies when the remote peer is defined for RSRB using IP encapsulation over a TCP connection. If no local acknowledgment is used, the remote peer can be defined for RSRB using direct encapsulation, RSRB using IP encapsulation over an FST connection, or RSRB using IP encapsulation over a TCP connection.

QLLC Conversion

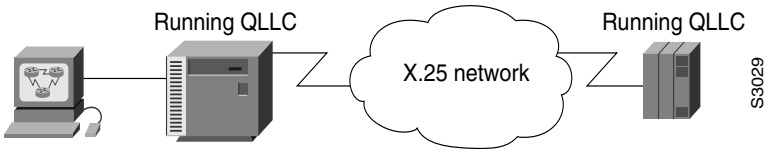
Qualified Logical Link Control (QLLC) is a data link protocol defined by IBM that allows SNA data to be transported across X.25 networks. (Although IBM has defined other protocols for transporting SNA traffic over an X.25 network, QLLC is the most widely used.) Figure 92 illustrates how QLLC conversion provides data link layer support for SNA communication.

Figure 92 SNA Data Link Layer Support



As shown in Figure 93, any devices in the SNA communication path that use X.25, whether end systems or intermediate systems, require a QLLC implementation.

Figure 93 SNA Devices Running QLLC

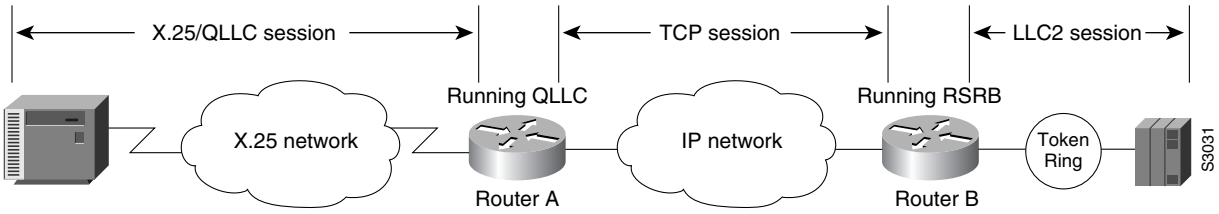


As shown in Figure 94, the QLLC conversion feature eliminates the need to install the X.25 software on local IBM equipment. A device attached locally to a Token Ring network can communicate through a router running the QLLC Conversion feature with a remote device attached to an X.25 network using QLLC. Typically, the locally attached device is an FEP, an AS 400, or a PS/2, and the remote device is a terminal controller or a PS/2. In this case, only the remote device needs an X.25 interface and the FEP can communicate with the terminal controller as if it were directly attached via a Token Ring network.

Figure 94 Router Running QLLC Conversion Feature

More elaborate configurations are possible. The router that implements QLLC conversion need not be on the same Token Ring network as the FEP. As shown in Figure 95, QLLC/LLC2 conversion is possible even when an intermediate IP WAN exists between the router connected to the X.25 network and the router connected to the Token Ring.

Figure 95 QLLC Conversion Running on a Router with an Intermediate IP Network

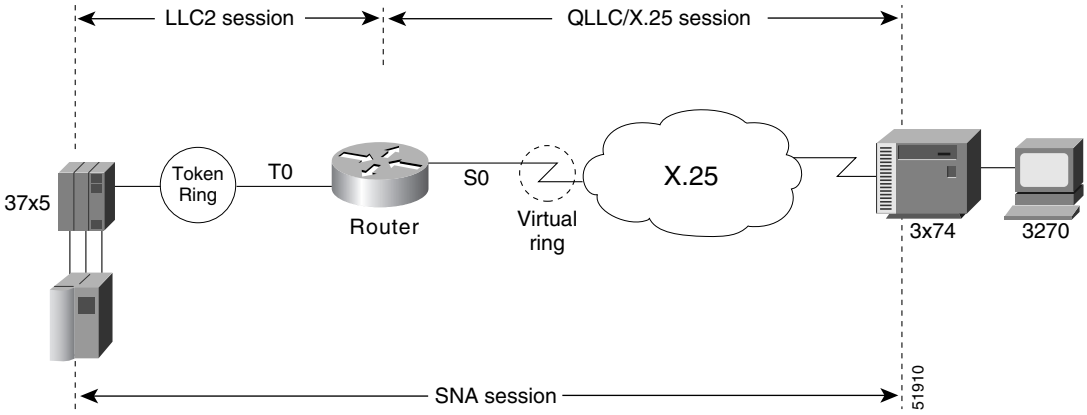


The Cisco Implementation of QLLC Conversion

SNA uses QLLC and X.25 as link layer protocols to provide a reliable connection. QLLC itself processes QLLC control packets. In a Token Ring environment, SNA uses LLC to provide a reliable connection. The LAN-to-X.25 (LNX) software provides a QLLC conversion function to translate between LLC and QLLC.

Figure 96 shows the simplest QLLC conversion topology: a single Token Ring device (for example, a 37x5 FEP) communicates with a single remote X.25 device (in this case a 3x74 cluster controller). In this example, a router connects the Token Ring network to the X.25 network.

Figure 96 QLLC Conversion Between a Single 37x5 and a Single 3x74

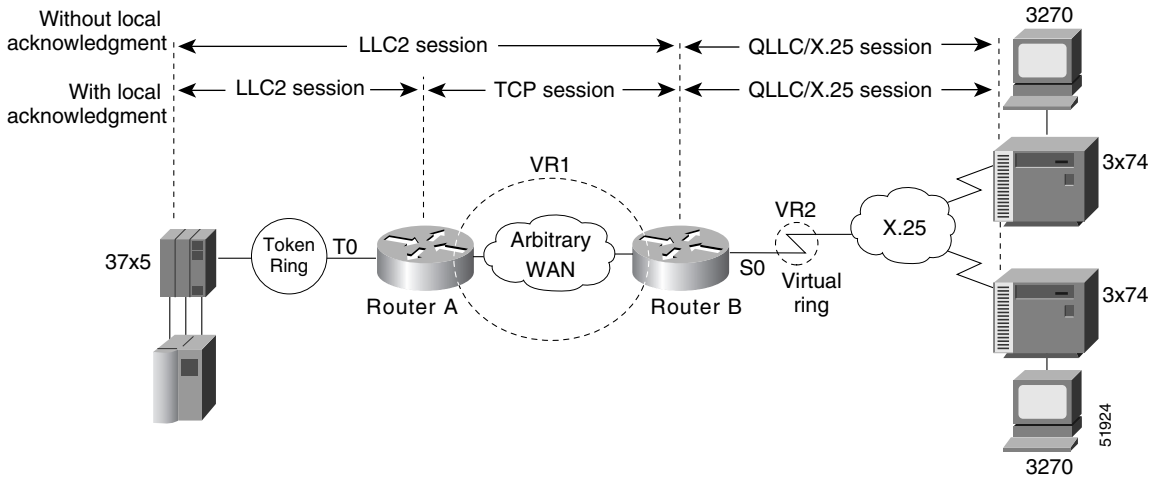


In Figure 96, each IBM end node has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were communicating over an X.25 network. This is accomplished by configuring the router's X.25 interface as a virtual Token Ring, so that the X.25 virtual circuit appears to the Token Ring device (and to the router itself) as if it were a Token Ring to which the remote X.25 device is attached.

Also in this figure, the LLC2 connection extends from the 37x5 FEP across the Token Ring network to the router. The QLLC/X.25 session extends from the router across the X.25 network to the 3x74 cluster controller. Only the SNA session extends across the Token Ring and X.25 networks to provide an end-to-end connection from the 37x5 FEP to the 3x74 cluster controller.

As Figure 97 shows, a router need not directly connect the two IBM end nodes; instead, some type of backbone WAN can connect them. Here, RSRB transports packets between Router A and Router B, while Router B performs all conversion between the LLC2 and X.25 protocols. Only the router attached to the serial line (Router B) needs to be configured for QLLC conversion. Both Router A and Router B are configured for normal RSRB.

Figure 97 QLLC Conversion Between a Single 37x5 and Multiple 3x74s across an Arbitrary WAN



How communication sessions are established over the communication link varies depending on whether or not LLC2 local acknowledgment has been configured on Router A's Token Ring interface. In both cases, the SNA session extends end-to-end and the QLLC/X.25 session extends from Router B to the 3x74 cluster controller. If LLC2 local acknowledgment has not been configured, the LLC2 session extends from the 37x5 FEP across the Token Ring network and the arbitrary WAN to Router B. In contrast, when LLC2 local acknowledgment has been configured, the LLC2 session extends from the 37x5 FEP Router A, where it is locally terminated. A TCP session is then used across the arbitrary WAN to Router B.

Comparing QLLC Conversion to SDLLC

Although the procedures you use to configure QLLC are similar to those used to configure SDLLC, there are structural and philosophical differences between the point-to-point links that SDLC uses and the multiplexed virtual circuits that X.25 uses.

The most significant structural difference between QLLC conversion and SDLLC is the addressing. To allow a device to use LLC2 to transfer data, both SDLLC and QLLC provide virtual MAC addresses. In SDLLC, the actual MAC address is built by combining the defined virtual MAC (whose last byte is 0x00) with the secondary address used on the SDLC link; in this way, SDLLC supports multidrop. In QLLC conversion, multidrop is meaningless, so the virtual MAC address represents just one session and is defined as part of the X.25 configuration. Because one physical X.25 interface can support many simultaneous connections for many different remote devices, you only need one physical link to the X.25 network. The different connections on different virtual circuits all use the same physical link.

The most significant difference between QLLC conversion and SDLLC is the fact that a typical SDLC/SDLLC operation uses a leased line. In SDLC, dial-up connections are possible, but the maximum data rate is limited. In QLLC, both switched virtual circuits (SVCs) and permanent virtual circuits (PVCs) are available, but the favored use is SVC. While the router maintains a permanent connection to the X.25 network, a remote device can use each SVC for some bounded period of time and then relinquish it for use by another device. Using a PVC is very much like using a leased line.

Table 3 shows how the QLLC commands correspond to the SDLLC commands.

Table 3 QLLC and SDLLC Command Comparison

QLLC Command	Analogous SDLLC Command
`qllc largest-packet`	`sdllc ring-largest-frame, sdllc sdlc-largest-frame`
`qllc partner`	`sdllc partner`
`qllc sap`	`sdllc sap`
`qllc srb, x25 map qllc, x25 pvc qllc`	`sdllc traddr`
`qllc xid`	`sdllc xid`
`source-bridge qllc-local-ack`	`source-bridge sdllc-local-ack`

Other Implementation Considerations

Consider the following when implementing QLLC conversion:

- To use the QLLC conversion feature, a router must have a physical link to an X.25 public data network (PDN). It must also have an SRB/RSRB path to an IBM FEP. This link could be a Token Ring or Ethernet interface, or even FDDI, if RSRB is being used.
- QLLC conversion can run on any router with at least one serial interface configured for X.25 communication and at least one other interface configured for SRB or RSRB.
- QLLC conversion security depends upon access control in SRB/RSRB and X.25 and upon XID validation.

You can configure DLSw+ for QLLC connectivity, which enables the following scenarios:

- Remote LAN-attached devices (physical units) or SDLC-attached devices can access an FEP or an AS/400 over an X.25 network.
- Remote X.25-attached SNA devices can access an FEP or an AS/400 over a Token Ring or over SDLC.

For information on configuring DLSw+ for QLLC conversion, refer to the “Configuring DLSw+” chapter.

You can configure DSPUs for QLLC. For more information on this configuration, refer to the “Configuring DSPU and SNA Service Point” chapter.

SNA FRAS

Using Frame Relay Access Support (FRAS), the Cisco IOS software allows branch SNA devices to connect directly to a central site front-end processor over a Frame Relay network. FRAS converts LAN or Synchronous Data-Link Control (SDLC) protocols to a Frame Relay format understood by the Network Control Program (NCP) that runs in an FEP. The Cisco IOS software and the NCP support two frame formats:

- RFC 1490 routed format for LLC2, specified in the FRF.3 Agreement from the Frame Relay Forum and known in NCP literature as Frame Relay Boundary Network Node (BNN) support. Support for this feature requires NCP 7.1 or higher.
- RFC 1490 802.5 source-route bridged format, known in NCP literature as Frame Relay Boundary Access Node (BAN) support. Support for this feature requires NCP 7.3 or higher.

Management service point support in FRAS allows the SNA network management application, NetView, to manage Cisco routers over the Frame Relay network as if it were an SNA downstream PU.

FRAS provides dial backup over RSRB in case the Frame Relay network is down. While the backup Public Switched Telephone Network (PSTN) is being used, the Frame Relay connection is tried periodically. As soon as the Frame Relay network is up, it will be used.

This section contains a brief overview of SNA FRAS which is described in the following topics:

RFC 1490 Routed Format for LLC2 (BNN)

RFC 1490 specifies a standard method of encapsulating multiprotocol traffic with data link (Level 2 of the OSI model) framing. The encapsulation for SNA data is specified in the FRF.3 Agreement.

The Frame Relay encapsulation method is based on the RFC 1490 frame format for "user-defined" protocols using Q.933 NLPID, as illustrated in Figure 98.

Figure 98 Frame Relay Encapsulation Based on RFC 1490

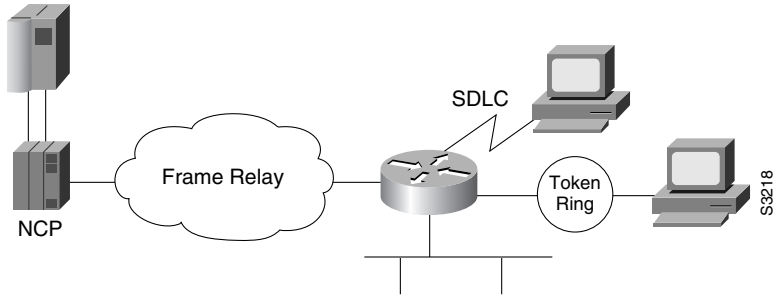
DLCI Q.922 address	Control 0x30	NLPID Q.933 0x08	L2 Protocol ID 0x4c (802.2) 0x08	L3 Protocol ID	DSAP SSAP	Control	F C S

51911

Note The protocol ID for SNA subarea FID4 is 0x81. The protocol ID for SNA subarea FID2 is 0x82. The protocol ID for APPN FID2 is 0x83.

FRAS allows the router acting as a FRAD to take advantage of the SNA BNN support for Frame Relay provided by ACF/NCP 7.1 and OS/400 V2R3. Downstream PU 2.0 and PU 2.1 devices can be attached to the router through SDLC, Token Ring, or Ethernet links. The router acting as a FRAD is connected to the Network Control Program (NCP) or AS/400 through a public or private Frame Relay network, as illustrated in Figure 99.

Figure 99 SNA BNN Support for Frame Relay



The frame format that communicates across the Frame Relay BNN link is defined in RFC 1490 for routed SNA traffic. From the perspective of the SNA host (for example an NCP or AS/400), the Frame Relay connection is defined as a switched resource similar to a Token Ring BNN link. Because the frame format does not include link addresses to allow the NCP to distinguish among SNA devices on the same

permanent virtual circuit, Cisco supports SAP multiplexing, which allows you to configure unique LLC2 SAPs for each downstream SNA device so that they can share a single permanent virtual circuit to an FEP.

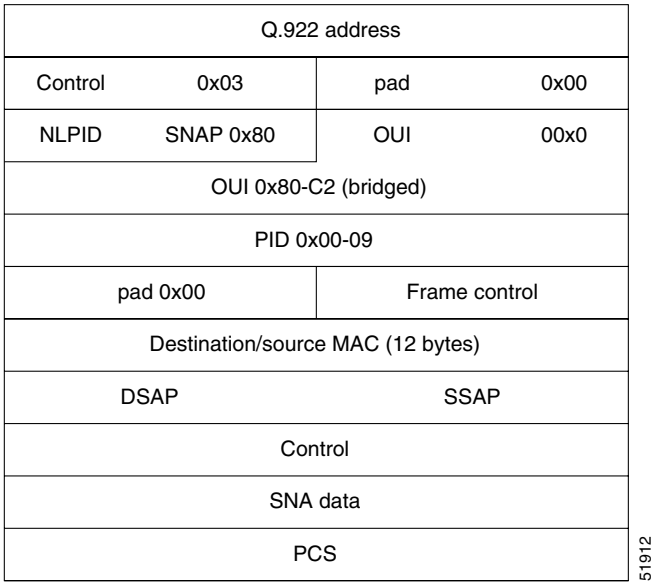
The Cisco IOS software is responsible for terminating the local data-link control frames (such as SDLC and Token Ring frames) and for modifying the data-link control frames to 802.2 compliant LLC frames. The LLC provides a reliable connection-oriented link layer transport required by SNA. (For example, 802.2 LLC is used to provide link layer acknowledgment, sequencing, and flow control.)

The Cisco IOS software encapsulates these 802.2 LLC frames according to the RFC 1490 format for SNA traffic. The frames are then forwarded to the SNA host on a Frame Relay PVC. In the reverse direction, the software is responsible for de-encapsulating the data from the Frame Relay PVC, and for generating and sending the appropriate local data-link control frames to the downstream devices.

RFC 1490 Bridged Format for LLC2 (BAN)

BAN provides functionality similar to BNN except that it uses a bridged frame format, as illustrated in Figure 100.

Figure 100 RFC 1490 Bridged Frame Format



Because it includes the MAC header information in every frame, BAN supports multiple SNA devices sharing a single permanent virtual circuit without requiring SAP multiplexing. BAN also supports load balancing across duplicate data-link connection identifiers to the same or different front-end processors at the data center to enhance overall availability. BAN works for devices attached by either Token Ring or Ethernet.

NCIA

Native Client Interface Architecture (NCIA) is a new software architecture introduced by Cisco to make accessing IBM SNA applications over routed internetworks more scalable and flexible. NCIA is a component of the Cisco IOS software. The architecture is intended to combine the benefits of the native SNA interface at end stations and mainframes with those of TCP/IP across the network backbone.

NCIA extends the use of the TCP/IP protocol all the way to the SNA end station. Because of the wide range of media supported by TCP/IP, including dialup telephone lines for remotely located users, NCIA makes multiprotocol access to corporate backbone networks much more flexible for SNA users.

NCIA allows SNA end stations such as PCs or workstations to encapsulate SNA traffic in TCP/IP, rather than requiring the traffic to travel through routers. The first phase of NCIA (NCIA I), used Cisco RSRB encapsulation. The current phase (NCIA Server) uses a new client/server model. NCIA Server is not backward compatible to NCIA I.

This section contains a brief overview of NCIA:

- NCIA I, page 26
- NCIA Server, page 26
- Advantages of the Client/Server Model, page 28

NCIA I

The Cisco NCIA server feature implements RFC 2114, *Data Link Switch Client Access Protocol*. Using the Cisco RSRB technology, NCIA I encapsulates the Token Ring traffic inside IP datagrams passed over a TCP connection between a router and a client. A virtual ring is created to allow the router to interconnect any client. The virtual ring acts as a logical Token Ring in the router, so that all the Token Rings connected to the router are treated as if they are all on the same Token Ring. The virtual ring is called a ring group. The ring group number is used just like a physical ring number and shows up in any route descriptors contained in packets being bridged. A ring group must be assigned a ring number that is unique throughout the network.

An NCIA I client acts as both an RSRB router and an end station. It must have a “fake” ring number and a “fake” bridge number so that it looks like an end station sitting on a real Token Ring. The fake ring and bridge numbers are visible to both the RSRB router and the NCIA client. The client must also have an LLC2 so that it can handle the LLC2 sessions.

NCIA Server

The NCIA Server feature extends the scalability of NCIA I, enhances its functionality, and provides support for both the installed base of RSRB routers and the growing number of DLSw+ routers. The NCIA Server feature includes the following enhancements:

- You do not need to configure a ring number on the client.
- You do not need to configure each client on the router.
- The MAC address can be dynamically assigned by the NCIA server running on the router.
- SNA is directly on top of TCP/IP; LLC2 is no longer required at end station.
- A client is a true end station, not a router peer.

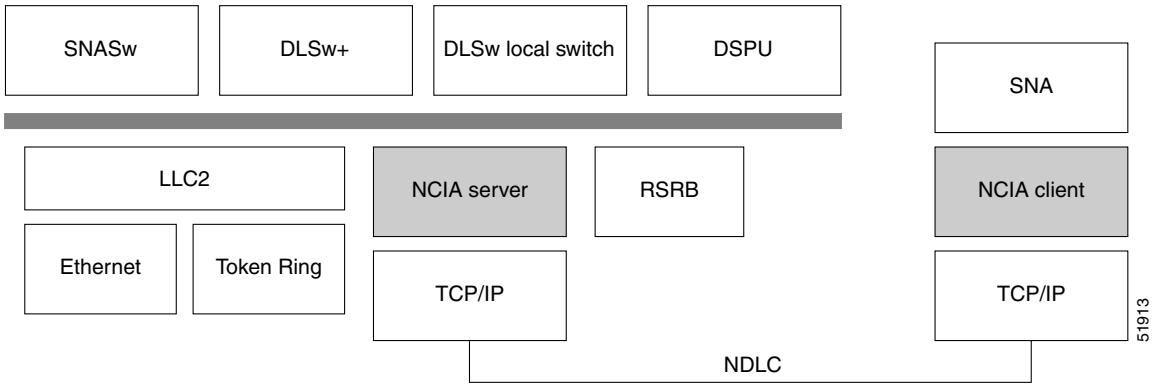
- The NCIA Server communicates with other components in router, such as RSRB, SNASw, DLSw+, and DSPU.
- Supports both connect-in and connect-out.
- The NCIA client/server model is independent of the upstream implementation.
- It is an efficient protocol between client and server.

NCIA Client/Server Model

The NCIA Server feature uses a client/server model (see Figure 101), where the NCIA server is a software module on a Cisco router and the NCIA client is a PC or workstation. The NCIA server performs two major functions:

- Establishes TCP to NCIA Data Link Control (NDLC) sessions with clients for the purpose of sending and receiving data.
- Uses the Cisco link services interface (CLSI) to communicate with other software modules in the router, such as SNASw, DLSw+, and DSPU, and acts as the data intermediary between them and NCIA clients. The NCIA server's role as an intermediary is transparent to the client.

Figure 101 NCIA Server Client/Server Model



NCIA Data Link Control (NDLC) is the protocol used between clients and servers. NDLC serves two purposes:

- Establishes the peer connection
- Establishes the circuit between the client and the server

The peer session must be established before an end-to-end circuit can be set up. During the set up period for the peer session, the MAC address representing a client is defined. The MAC address can be defined by the client or by the server when the client does not have a MAC address.

The NCIA Server feature supports connect-in and connect-out (from the server's perspective), but connect-out is not supported if the client station does not listen for the incoming connection. For a server to connect-out, clients must connect to the server first. After registering itself by providing its own MAC address, the client can then optionally disconnect from the server. When a server receives an explorer, and its destination MAC address is registered, an NCIA server will connect to that client if it is not connected. For NetBIOS explorers (addressed to functional address 0xC00000000080), the TCP session must remain up so that the server can broadcast the explorers to the client. If the TCP session is down, the server will not send the NetBIOS explorers to a client, even when the client is registered.

After the peer session has been established, the NDLC protocol establishes the circuit between the client and server. This circuit is used to transfer end-user data between the client and the server. Because the client and its target station are not on the same transport, they cannot form a direct, end-to-end circuit. Each client must form a circuit between the client and server, and the server must form another circuit between the server and the target station. The server links those two circuits to form an end-to-end circuit. The server acts as a mediator between the client and the target station so that packets can be transferred between them.

In the NCIA server only peer keepalive is maintained. There is no keepalive at circuit level.

The NCIA server acts as a data-link provider, like Token Ring or Ethernet, in the router. It uses CLSI to communicate with other software modules, just as other data-link providers do. The network administrator configures the router to communicate with specific modules. For data-link users, such as SNASw, DLSw+, and DSPU, the NCIA server can interface to them directly. For other data-link providers, the NCIA server must go through a DLSw+ local peer to communicate with them. The DLSw+ local peer passes packets back and forth among different data-link providers.

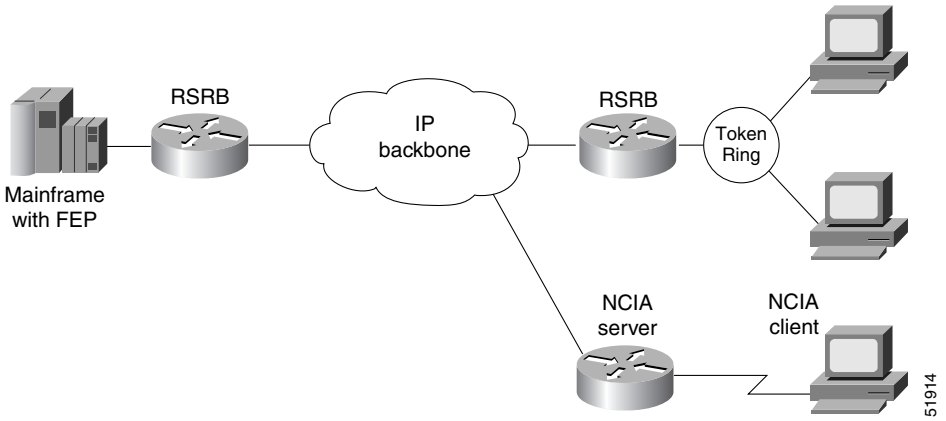
Advantages of the Client/Server Model

The client/server model used in the NCIA Server feature extends the scalability of NCIA. In addition, it provides support for both the installed base of RSRB routers and the growing number of DLSw+ routers.

Extended Scalability

The client/server model minimizes the number of central site RSRB or DLSw+ peer connections required to support a large network of NCIA clients (see Figure 102). Rather than each client having a peer connection to a central site router, the clients attach to an IP backbone through an NCIA server that, in turn, has a single peer connection to a central site router. This scheme can greatly reduce the number of central site peer connections required. For example, in a network with 1000 clients and 10 NCIA servers, there would be only 10 central site peer connections. Note that there would still be 1000 LLC2 connections that must be locally acknowledged at the central site router, but this can easily be handled in a single central site router. When the number of LLC2 connections (or the number of clients) is in the tens of thousands, NCIA servers can take advantage of downstream PU concentration to minimize the number of LLC2 connections that must be supported by the central site routers.

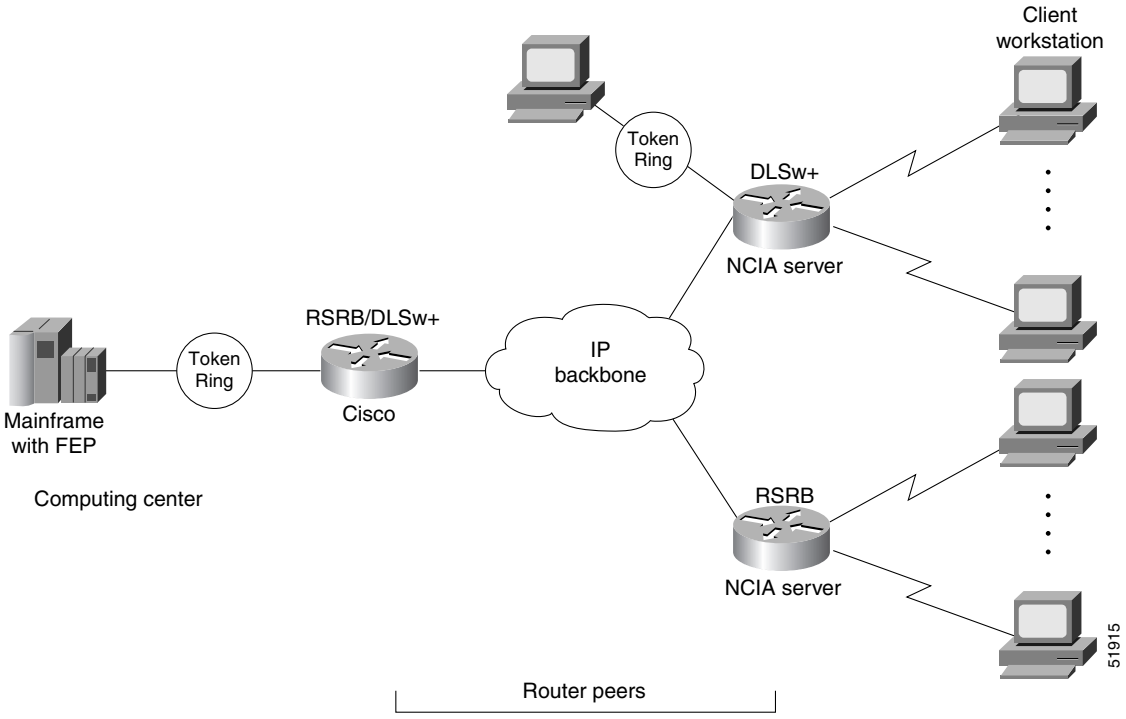
Figure 102 NCIA Server Provides Extended Scalability to Support Large Networks



Migration Support

Using a client/server model allows the NCIA Server feature to be independent of the upstream implementation, allowing it to be implemented in a network that is still using RSRB and in a DLSw+ network. It also greatly simplifies migration from RSRB to DLSw+, because it requires no changes at the client. A single NCIA server can support either approach (but not both). As Figure 103 illustrates, a central site router can support RSRB and DLSw+ concurrently, allowing a portion of the NCIA servers to communicate using RSRB and another portion to communicate using DLSw+.

Figure 103 NCIA Server Provides Independence from the Upstream Network Implementation

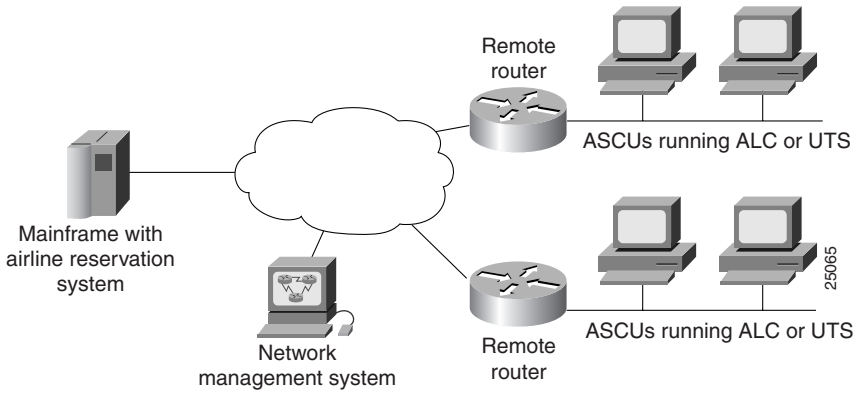


ALPS

The Airline Product Set (ALPS) is a tunneling mechanism that transports airline protocol data across a TCP/IP network to a mainframe. ALPS provides connectivity between agent set control units (ASCUs) and a mainframe host that runs the airline reservation system.

Figure 104 shows the basic ALPS topology and the protocols implemented in the feature. Three major components provide the end-to-end transportation of airline protocol traffic across the network: the P1024B Airline Control (ALC) or P1024C (UTS) protocol, the TCP/IP-based MATIP protocol conversion, and the TCP/IP access to the mainframe.

Figure 104 ALPS Architecture



The Cisco ALPS feature provides an end-to-end solution for airlines and central reservation systems.The ALPS feature is integrated in the Cisco IOS software and allows airlines to replace their existing hardware and software with Cisco routers. For customers who already use Cisco routers, this feature allows them to consolidate networking overhead and functionality.

DSPU and SNA Service Point

Downstream physical unit (DSPU) is a software feature that enables the router to function as a PU concentrator for SNA PU type 2 nodes. PU concentration at the device simplifies the task of PU definition at the upstream host while providing additional flexibility and mobility for downstream PU devices.

The DSPU feature allows you to define downstream PU type 2 devices in the Cisco IOS software. DSPU reduces the complexity of host configuration by letting you replace multiple PU definitions that represent each downstream device with one PU definition that represents the router.

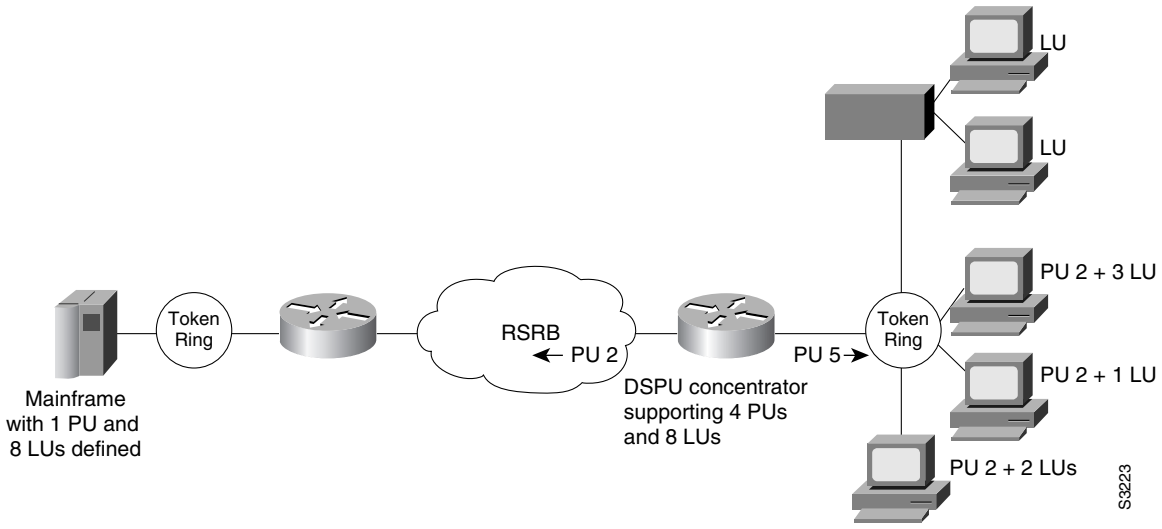
Because you define the downstream PUs at the router rather than the host, you isolate the host from changes in the downstream network topology. Therefore you can insert and remove downstream PUs from the network without making any changes on the host.

The concentration of downstream PUs at the router also reduces network traffic on the WAN by limiting the number of sessions that must be established and maintained with the host. The termination of downstream sessions at the router ensures that idle session traffic does not appear on the WAN.

SNA service point support in the Cisco IOS software assumes that NetView or an equivalent product is available at the SNA host. The user interacts with the network management feature in the router and at the SNA host. In the Cisco IOS software, you can configure the host connection and show the status of this connection. At the SNA host, you can use the NetView operator's console to view alerts and to send and receive Cisco syntax commands to the Cisco device.

Figure 105 shows a router functioning as a DSPU concentrator.

Figure 105 Router Acting as a DSPU Concentrator

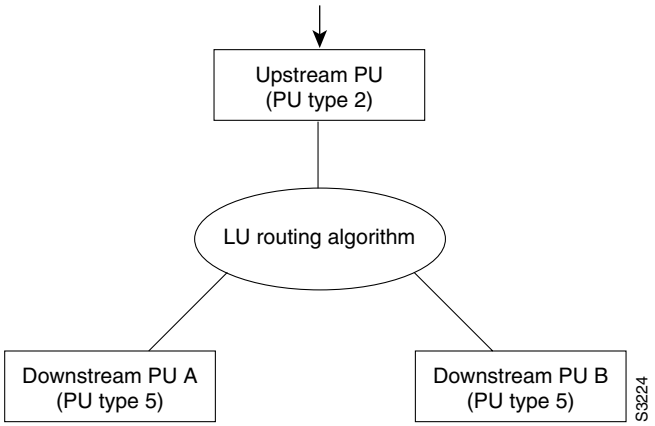


Typically, a router establishes one or more upstream connections with one or more hosts and many downstream connections with PU type 2 devices. From an SNA perspective, the router appears as a PU type 2 device to the upstream host and assumes the role of a system services control point (SSCP) appearing as a PU type 5 device to its downstream PUs.

The SSCP sessions established between the router and its upstream host are completely independent of the SSCP sessions established between the router and its downstream PUs. SNA traffic is routed at a logical unit (LU) level using a routing algorithm that maps downstream LUs onto upstream LUs.

Figure 106 illustrates the SNA perspective of DSPU.

Figure 106 SNA Perspective of DSPU



SNA Switching Services

Note SNA Switching Services functionality supersedes all functionality previously available in the APPN feature in the Cisco IOS software. SNASw configuration will not accept the previous APPN configuration commands. Previous APPN users should use this chapter to configure APPN functionality using the new SNASw commands.

SNASw provides an easier way to design and implement networks with SNA routing requirements. Previously, this network design was accomplished using APPN with full network node (NN) support in the Cisco router. This type of support provided the SNA routing functionality needed, but was inconsistent with the trends in Enterprise networks today. The corporate intranet is replacing the SNA WAN. Enterprises are replacing their traditional SNA network with an IP infrastructure that supports traffic from a variety of clients, using a variety of protocols, requiring access to applications on a variety of platforms, including SNA applications on Enterprise servers.

While SNA routing is still required when multiple servers must be accessed, the number of nodes required to perform this function is decreasing as the IP infrastructure grows and as the amount of native SNA traffic in the network decreases.

SNASw enables an enterprise to develop their IP infrastructure, while meeting SNA routing requirements.

The number of NNs in the network and the amount of broadcast traffic are reduced. Configuration is simplified, and SNA data traffic can be transported within the IP infrastructure. The following features provide this functionality:

- HPR Capable SNA Routing Services, page 34
- Branch Extender, page 34
- Enterprise Extender (HPR/IP), page 35
- Usability Features, page 36
- Management Enhancements, page 37
- LAN and IP-Focused Connection Types, page 38

Benefits of SNASw

SNASw provides the following benefits:

- Scalable APPN Networks, page 33
- IP Infrastructure Support, page 33
- Reduced Configuration Requirements, page 33
- Network Design Simplicity, page 33
- Improved Availability, page 33
- Increased Management Capabilities, page 33
- Architectural Compliance, page 33

Scalable APPN Networks

With the Branch Extender (BEX) function, the number of network nodes and the amount of broadcast traffic are reduced.

IP Infrastructure Support

Limiting SNASw routers to the data center and using the BEX function eliminates SNA broadcasts from the IP network. With Enterprise Extender (EE), SNA traffic is routed using the IP routing infrastructure while maintaining end-to-end SNA services.

Reduced Configuration Requirements

By eliminating NNs and using the BEX function, configuration tasks are minimized. Additionally, Cisco has enhanced its auto-configuration capability to eliminate previously required commands.

Network Design Simplicity

By placing all SNA routers in the data center, few SNA routers are required, and they can be easily configured using virtually identical configurations.

Improved Availability

By adding Cisco-unique capabilities to connect-out and distribute traffic across multiple ports, access to resources is improved and traffic can be distributed across multiple ports. Additionally, by supporting the newest HPR Adaptive Rate-Based (ARB) flow control algorithm, bandwidth management for SNA traffic is improved.

Increased Management Capabilities

Two new traces, interprocess and data-link, provide an easier way to view SNASw activity. The APPN Trap MIB allows the user to notify the operator in event of a debilitating problem. Console message archiving provides better tracking of network activity. The ability to format traces in a format so that they are readable by other management products simplify network management because results are more readily available.

Architectural Compliance

Even though SNASw is easier to use and SNASw networks are easier to design, SNASw interfaces with SNA implementations on the market: upstream NNs, end nodes (ENs), low-entry networking (LEN) nodes and PU 2.0. It also provides full DLUR support to allow dependent PU and LU traffic to flow over the APPN network to SNA data hosts.

HPR Capable SNA Routing Services

SNASw provides the following SNA routing functions:

- Routes SNA sessions between clients and target SNA data hosts.
- Controls SNA traffic in a multiprotocol environment in conjunction with other Cisco IOS quality of service (QoS) features.
- Supports networks with a high proportion of SNA traffic and multiple enterprise servers, especially those that continue to support the traditional SNA endstation platform and new client types.
- Supports all types of SNA application traffic including traditional 3270 and peer LU 6.2.
- Supports an OS/390 Parallel Sysplex configuration, working in conjunction with the IBM Communications Server for S/390 (formerly VTAM) and the MVS Workload Manager, to provide higher availability in the data center using the High Performance Routing (HPR) feature.
- Supports System Services Control Point (SSCP) services to downstream SNA devices using the Dependent LU Requester (DLUR) feature.
- Provides dynamic link connectivity using connection networks (CNs), which eliminates much of the configuration required in networks with numerous data hosts.

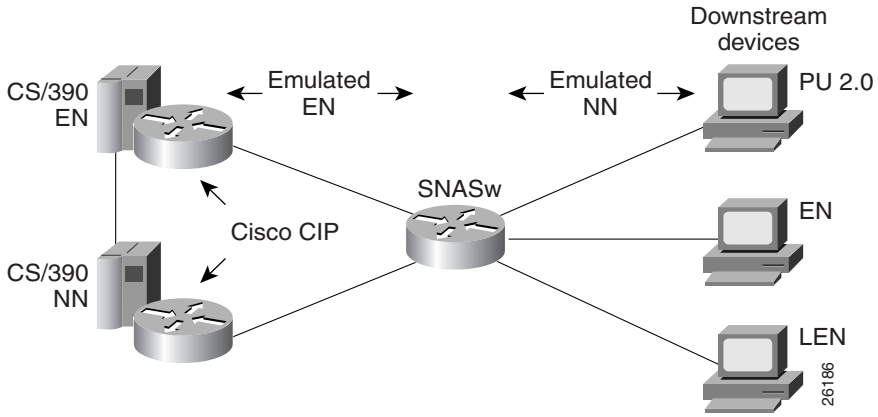
Branch Extender

The BEX function enhances scalability and reliability of SNA routing nodes by eliminating topology updates and broadcast directory storms that can cause network instability. BEX appears as an NN to downstream EN, LEN node, and PU devices, while also appearing as an EN to upstream devices. The BEX function eliminates APPN topology and APPN broadcast search flows between SNASw nodes and the SNA data hosts in the network. This feature is key to providing a reliable turn-key installation because the network administrator no longer needs to develop in-depth knowledge of the level and characteristics of broadcast directory search and topology update traffic in the network. Such knowledge and analysis was commonly required to build successful networks utilizing NN technology without BEX.

SNA Switching Services enables BEX functionality by default. SNASw treats all defined links as BEX “uplinks” and all dynamic links created by stations connecting into SNASw as BEX “downlinks.” No specific configuration is necessary to enable BEX functionality.

Figure 107 illustrates the BEX functionality.

Figure 107 BEX Functionality

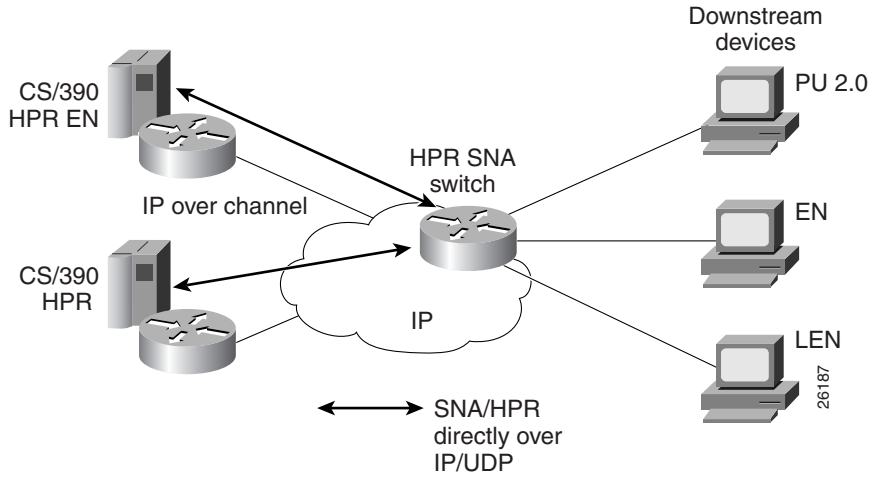


Enterprise Extender (HPR/IP)

SNASw also supports the EE function. EE offers SNA HPR support directly over IP networks. EE also uses connectionless User Datagram Protocol (UDP) transport. SNA COS and transmission priority are maintained by mapping the transmission priority to the IP precedence and by mapping transmission priority to separate UDP port numbers, allowing the IP network to be configured based on these elements. The Cisco IP prioritization technologies, such as weighted fair queueing (WFQ), prioritize the traffic through the IP network. EE support on the IBM Communications Server for S/390 allows users to build highly reliable SNA routed networks that run natively over an IP infrastructure directly to the Enterprise servers. These network designs reduce points of failure in the network and provide reliable SNA networks.

Figure 108 illustrates the EE functionality.

Figure 108 EE Functionality



Usability Features

SNASw contains the following usability features designed to make SNA networks easier to design and maintain:

Dynamic CP Name Generation Support

When scaling the SNASw function to hundreds or thousands of nodes, many network administrators find that defining a unique control point (CP) name on each node provides unnecessary configuration overhead. Dynamic CP name generation offers the ability to use the Cisco IOS hostname as the SNA CP name or to generate a CP name from an IP address. These facilities reuse one SNASw configuration across many routers and eliminate the specific configuration coordination previously required to configure a unique CP name for each SNA node in the network. Administrators can still explicitly configure the CP name within the SNASw configuration.

Dynamic SNA BTU Size

Most SNA node implementations require specific tuning of the SNA basic transmit unit (BTU) in the configuration. SNASw analyzes the interface maximum transfer units (MTUs) of the interfaces it uses and dynamically assigns the best MTU values for that specific port. For served dependent PU 2.0 devices, SNASw uses the downstream MAXDATA value from the host and dynamically sets the SNA BTU for that device to the MAXDATA value.

DLUR Connect-Out

SNASw can receive connect-out instructions from the IBM Communications Server for S/390. This function allows the system to dynamically connect-out to devices that are configured on the host with the appropriate connect-out definitions. This feature allows connectivity to SNA devices in the network that were traditionally configured for connect-out from the host.

Note DLUR connect-out can be performed over any supported data-link type.

Responsive Mode Adaptive Rate-Based Flow Control

Early HPR implementations failed to perform well in environments subject to packet loss (for example, Frame Relay, IP transport) and performed poorly when combined with other protocols in multiprotocol networks. SNASw implements the second-generation HPR flow control architecture, called Responsive

Mode ARB architecture. Responsive Mode ARB addresses all the drawbacks of the earlier ARB implementation, providing faster ramp-up, better tolerance of lost frames, and better tolerance of multiprotocol traffic.

User-Settable Port Limits

SNASw offers full control over the number of devices supported by a specific port. The max-links configuration on the SNASw port controls the number of devices that are served by this port. When the max-links limit is reached, SNASw no longer responds to test frames attempting to establish new connections. SNASw allows load sharing among different SNASw nodes that offer service to the same SNA MAC addresses.

Management Enhancements

SNASw contains the following enhanced tools for managing SNA networks:

- Console Message Archiving, page 37
- Data-Link Tracing, page 37
- Interprocess Signal Tracing, page 37
- MIB Support for Advanced Network Management Awareness, page 38

Console Message Archiving

Messages issued by SNASw are archived in a buffer log that is queried and searched on the console or transferred to a file server for analysis. Each message has a single line that identifies the nature of the event that occurred. The buffer log also maintains more detailed information about the message issued.

Data-Link Tracing

SNA frames entering or leaving SNASw are traced to the console or to a cyclic buffer. These frames are analyzed at the router or transferred to a file server for analysis. The trace is sent to a file server in a SNA-formatted text file or in binary format readable by existing traffic analysis applications.

Interprocess Signal Tracing

The SNASw internal information is traced in binary form, offering valuable detailed internal information to Cisco support personnel. This information helps diagnose suspected defects in SNASw.

MIB Support for Advanced Network Management Awareness

SNASw supports the following Management Information Bases (MIBs):

- IETF draft standard DLUR MIB (RFC 2232), which defines objects for monitoring and controlling network devices with DLUR (Dependent LU Requester) capabilities.
- IETF draft standard APPN MIB (RFC 2455), which defines objects for monitoring and controlling network devices with Advanced Peer-to-Peer Networking (APPN) capabilities.
- APPN Traps MIB (RFC 2456), which defines objects for receiving notifications from network devices with APPN and DLUR capabilities. This MIB proactively send traps with information about changes in SNA resource status. This implementation reduces the frequency of SNMP polling necessary to manage SNA devices in the network.

The CiscoWorks Blue Maps application retrieves relevant SNASw data from these MIBs and displays it in a manner that simplifies and speeds up problem isolation and resolution.

LAN and IP-Focused Connection Types

SNASw supports several connection types to serve all SNA connectivity options, including the following types:

- Token Ring, Ethernet, and FDDI, page 38
- Virtual Token Ring, page 38
- Virtual Data-Link Control, page 39
- Native IP Data-Link Control (HPR/IP), page 39

Token Ring, Ethernet, and FDDI

SNASw natively supports connectivity to Token Ring, Ethernet, and FDDI networks. In this configuration mode, the MAC address used by SNASw is the local configured or default MAC address of the interface.

Virtual Token Ring

Using virtual Token Ring allows SNASw access to SRB, which allows the following configuration:

- Attachment to Local LANs, page 38
- Connection to Frame Relay Transport Technologies, page 39
- Connection to Channel Interface Processor and Channel Port Adapter, page 39

Attachment to Local LANs

Virtual Token Ring allows you to connect to local LAN media through SRB technology. Because there is no limit to the number of virtual Token Ring interfaces that can connect to a specific LAN, this technology allows configuration of multiple MAC addresses, which respond to SNA requests over the

same LAN. When using native LAN support, SNASw responds only to requests that target the MAC address configured on the local interface. Virtual Token Ring and SRB allow SNASw to respond to multiple MAC addresses over the same physical interface.

Connection to Frame Relay Transport Technologies

Virtual Token Ring and SRB connect SNASw to a SNA Frame Relay infrastructure. FRAS host and SRB Frame Relay are configured to connect virtual Token Ring interfaces that offer SNASw support for Frame Relay boundary access node (BAN) or boundary network node (BNN) technology.

Connection to Channel Interface Processor and Channel Port Adapter

Virtual Token Ring and SRB can be used to connect SNASw to the Channel Interface Processor (CIP) or Channel Port Adapter (CPA) in routers that support those interfaces.

Virtual Data-Link Control

SNASw uses Virtual Data-Link Control (VDLC) to connect to DLSw+ transport and local switching technologies. VDLC is used for a number of connectivity options, including the following two:

- Transport over DLSw+ Supported Media, page 39
- DLC Switching Support for Access to SDLC and QLLC, page 39

Transport over DLSw+ Supported Media

Using VDLC, SNASw gains full access to the DLSw+ transport facilities, including DLSw+ transport over IP networks, DLSw+ transport over direct interfaces, and DLSw+ support of direct Frame Relay encapsulation (without using IP).

DLC Switching Support for Access to SDLC and QLLC

Through VDLC, SNASw gains access to devices connecting through SDLC and QLLC. This access allows devices connecting through SDLC and QLLC access to SNASw.

Native IP Data-Link Control (HPR/IP)

SNASw support for the EE function provides direct HPR over UDP connectivity. This support is configured for any interface that has a configured IP address. HPR/IP uses the interface IP address as the source address for IP traffic originating from this node.

Cisco Transaction Connection

This section contains the following topics:

- CTRC and CICS, page 40
- CTRC and DB2, page 41
- Benefits of CTRC, page 42

The CTRC software feature provides the following functionality:

- CTRC allows Cisco routers to use the intersystem communication (ISC) protocol to provide a gateway between Customer Information Control System (CICS) clients (also known as common clients) running under Windows or UNIX on TCP/IP networks and CICS online transaction monitoring systems on IBM hosts.
- CTRC supports two interfaces to common clients: the Extended Call Interface (ECI), which lets non-CICS client programs call CICS transactions, and the Extended Presentation Interface (EPI), which lets distributed applications call CICS transactions that were originally accessed via 3270 terminals.
- CTRC supports the ability to configure routes for CICS transaction. Each transaction can be routed to a specific CICS region.
- In addition to its CICS-related functionality, CTRC includes the feature previously known as Cisco Database Connection (CDBC), which allows Cisco routers to use IBM's distributed relational database architecture (DRDA) protocol to provide a gateway between client workstations running Open DataBase Connectivity (ODBC) compliant applications on TCP/IP networks and IBM DB2 databases on SNA networks. ODBC is a call-level interface developed by Microsoft Corporation that allows a single application to access database management systems from different vendors using a single interface. SNA is a large, complex, feature-rich network architecture developed by IBM.
- CTRC adds support for TCP/IP passthrough, allowing the use of a TCP/IP network, rather than a SNA network, between a Cisco router and a DB2 database if the database version supports direct TCP/IP access.
- To match functionality provided in DRDA over TCP/IP, CTRC adds support for Password Expiration Management (PEM) in SNA networks where PEM is supported.
- CTRC supports the following MIBs:
  - CISCO-DATABASE-CONNECTION-MIB.my - 93
  - CISCO-TRANSACTION-CONNECTION-MIB.my - 144

  For descriptions of supported MIBs and how to use MIBs, see the Cisco MIB website on Cisco.com.

CTRC and CICS

CTRC is a Cisco IOS software feature that is available in two environments:

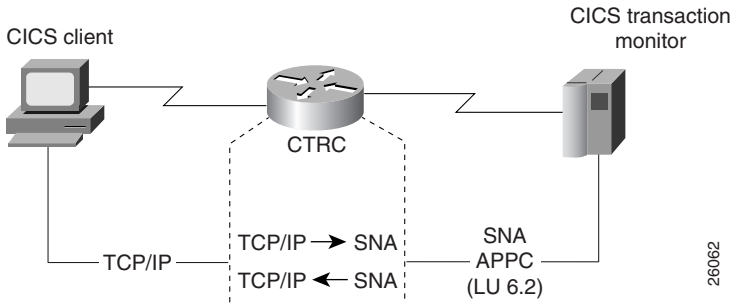
- CICS
- DB2

When a router is configured to use CTRC for communications with CICS systems, the router converts ISC packets over TCP/IP to ISC packets over Advanced Program-to-Program Communications (APPC) LU 6.2 and then routes them to the appropriate CICS region. CTRC converts CICS client messages received via TCP/IP to SNA messages and uses Cisco SNA Switching Services to send them to the host.

CTRC runs as a TCP/IP daemon on the router, accepting ISC client connections over TCP/IP. When a client connects to a CICS region on an IBM mainframe host, CTRC allocates an APPC conversation over SNA to an IBM server and acts as a gateway between ISC over TCP/IP and ISC over APPC.

Figure 109 illustrates how CTRC lets CICS client applications on TCP/IP networks interact with CICS transaction monitoring systems on IBM hosts.

Figure 109 Cisco Router Configured with the CTRC Feature for CICS Communications



CTRC and DB2

CTRC enables Cisco routers to implement IBM's DRDA over TCP/IP. The Cisco router with CTRC exists in the TCP/IP network, and clients use a CTRC IP address and port on the router to connect to the IBM host system that exists in either an SNA network or a TCP/IP network.

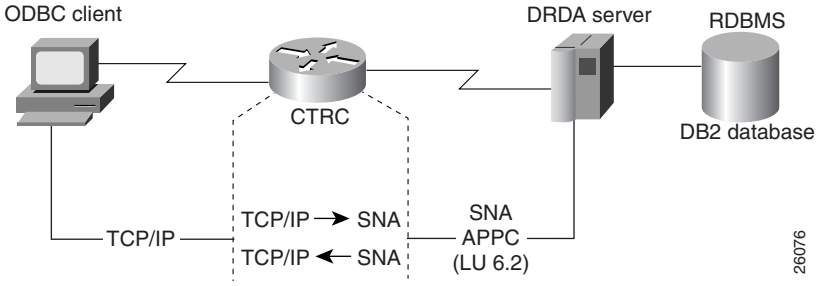
When CTRC is appropriately configured on a router, client-based ODBC applications can connect to the following IBM D2 relational databases:

- DB2 for OS/390 (MVS)
- DB2 for Virtual Machine (VM) (SQL/DS)
- DB2 for Virtual Storage Extended (VSE) (SQL/DS)
- DB2 for OS/400
- DB2 Universal Database (UNIX, Windows, OS/2)

For an SNA host connection, the router with CTRC converts DRDA packets over TCP/IP to DRDA packets over (APPC LU 6.2) and then routes them to DB2 databases. CTRC runs as a TCP/IP daemon on the router, accepting DRDA client connections over TCP/IP. When a client connects to the database on an IBM mainframe host, CTRC allocates an APPC conversation over SNA to an IBM server, and acts as a gateway between DRDA over TCP/IP and DRDA over APPC.

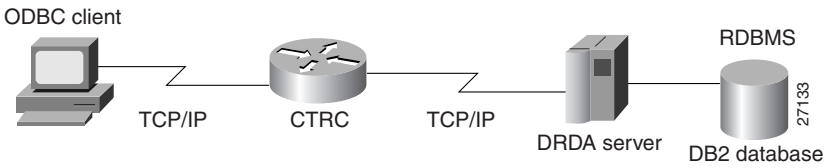
Figure 110 illustrates how the Cisco router configured with the CTRC feature enables the exchange of database information between ODBC client applications running DRDA in a TCP/IP network and a DRDA-based IBM system that accesses DB2 relational data.

Figure 110 Cisco Router Configured with the CTRC Feature for DB2 Communications (SNA Host Network)



For a TCP/IP host connection, the router with CTRC routes the DRDA packets over TCP/IP without protocol changes. To use this TCP/IP passthrough feature of CTRC, the host database version must support direct TCP/IP access. Figure 111 illustrates such a configuration.

Figure 111 Cisco Router Configured with the CTRC Feature for DB2 Communications (TCP/IP Host Network)



When configured for DB2 communications on a router, the CTRC feature enables desktop applications to access data in remote databases located on IBM hosts. CTRC receives database access messages from the client over a TCP/IP link. CTRC either converts the messages to SNA and sends them to the host using APPC services provided by the Cisco SNA Switching Services, or routes the client messages to the TCP/IP-enabled host without protocol changes.

Benefits of CTRC

CTRC provides TCP/IP end-users and servers with fast, reliable, and secure access to IBM DB2 databases using the SNA protocol. CTRC replaces expensive and hard to manage UNIX and NT gateways for database access.

CTRC lets Windows or UNIX client applications call CICS transactions without requiring changes to the client or host software.

In addition, CTRC provides Cisco 7200 and 7500 series routers with the functionality previously available in CDBC, which gives ODBC client applications access to data in DB2 databases.

CMCC Adapter Hardware

A CMCC adapter is installed in a Cisco router to provide IBM channel attachment from the router to a mainframe host. The Cisco family of CMCC adapters consists of two basic types of adapters:

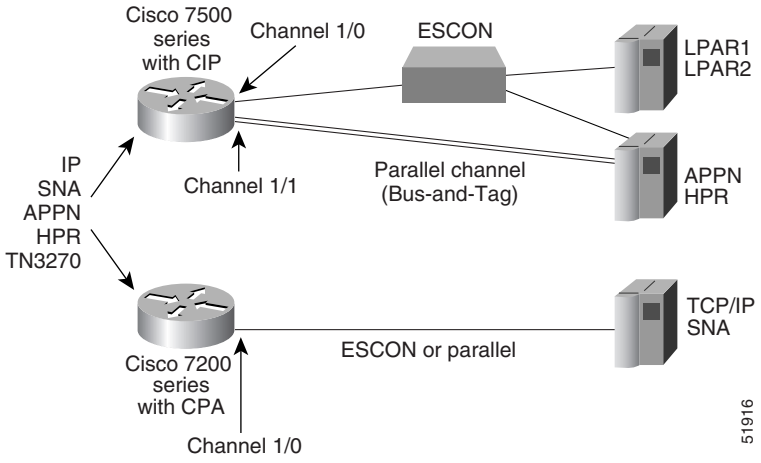
- Channel Interface Processor (CIP)—Installed on Cisco 7000 with RSP7000 and Cisco 7500 series routers
- Channel Port Adapter (CPA)—Installed on Cisco 7200 series routers

Each type of adapter (CIP or CPA) supports both ESCON and parallel channel attachment to the host and can eliminate the need for a separate FEP.

All CMCC adapters support the full range of channel software applications available in the Cisco IOS software including support for the Common Link Access to Workstation (CLAW) protocol, TCP/IP offload, IP host backup, Cisco SNA (CSNA), Cisco Multipath Channel (CMPC), Cisco Multipath Channel+ (CMPC+), and the TN3270 server.

Figure 112 shows the type of channel connections and environments supported by the CMCC adapters.

Figure 112 Cisco Mainframe Channel Connection Adapters



Channel Interface Processor

The CIP for the Cisco 7000 with RSP7000 and Cisco 7500 series routers is designed for high-end network environments that demand high-performance, high-port density, and high-capacity solutions.

The CIP provides support for IBM ESCON and bus-and-tag parallel channel attachment using the following types of interfaces:

- ESCON Channel Adapter (ECA)
- Parallel Channel Adapter (PCA)

A single CIP can support up to two physical channel interfaces in any combination of either PCA or ECA. Each CIP is pre-configured with the appropriate channel adapters at manufacturing time.

The Cisco 7000 with RSP7000 and Cisco 7500 series routers support online insertion and removal (OIR), which allows you to install or remove CIPs while the system is operating.

Channel Port Adapter

The CPA is available for the Cisco 7200 series routers. The CPA expands the value of the Cisco IBM channel solution by providing channel connectivity to mid-range mainframe configurations.

The CPA is a standard, single-width port adapter that provides support for IBM ESCON and bus-and-tag parallel channel attachment using the following types of interfaces:

- ESCON Channel Port Adapter (ECPA)
- Parallel Channel Port Adapter (PCPA)

Each CPA provides a single channel interface (with a single I/O connector) for Cisco 7200 series routers. In some situations, this eliminates the need for a separate front-end processor (FEP).

The only differences between CMCC software applications running on the CIP and a CPA are performance and capacity. The performance difference is based upon differences in the internal bus architecture of a CIP and a CPA, and the capacity difference is based on the difference in maximum memory configurations (128 MB for CIP and 32 MB for CPA). For more information about differences between the CIP and CPA, see the "Differences Between the CIP and CPA" section on page 44.

The Cisco 7200 series router supports online insertion and removal (OIR), which allows you to install or remove port adapters while the system is operating.

Note In this chapter, references to Channel Port Adapter (CPA) correspond to both the ECPA and the PCPA. Refer to the *Cisco 7200 Series Port Adapter Hardware Configuration Guidelines* publication for more details.

ESCON Channel Port Adapter

An ECPA is classified as a high-speed port adapter providing a single ESCON physical channel interface. Current Cisco 7200 configuration guidelines recommend using no more than three high-speed port adapters in a single Cisco 7200 router.

Parallel Channel Port Adapter

A PCPA provides a single parallel channel physical interface supporting 3.0 or 4.5 Mbps data transfer rates.

Differences Between the CIP and CPA

Table 4 illustrates the differences between the CMCC adapters.

Table 4 Differences Between the CIP and the CPA

Product Differences	CIP	ECPA	PCPA
Router platform	Cisco 7500 Cisco 7000 with RSP7000	Cisco 7200	Cisco 7200
Channel interfaces	ESCON Parallel	ESCON	Parallel
Maximum number of interfaces	2	1	1
Maximum memory	128 MB	32 MB	32 MB
Cisco IOS release support	Cisco IOS Release 10.2 and later	Cisco IOS Release 11.3(3)T and later	Cisco IOS Release 11.3(3)T and later
Virtual port number	2	0	0
Channel interface state tracking (HSRP, SNMP alerts)	Yes	Disabled—Use the **state-tracks-signal** command to enable	Disabled—Use the **state-tracks-signal** command to enable

Supported Environments

The Cisco IOS software supports the following environments and features on the CMCC adapters:

- TCP/IP Environments—CLAW, TCP/IP offload, IP host backup, CMPC+, and TN3270 server features
- SNA and APPN Environments—CSNA, CMPC, and TN3270 server features

CMCC Adapter Features for TCP/IP Environments

The Cisco IOS software supports the following features for CMCC adapters in TCP/IP environments:

- Common Link Access to Workstation, page 45
- TCP/IP Offload, page 45
- IP Host Backup, page 46
- Cisco Multipath Channel+, page 46
- TN3270 Server, page 48

Common Link Access to Workstation

To transport data between the mainframe and a CMCC adapter in TCP/IP environments, Cisco IOS software implements the CLAW channel protocol. Each CLAW connection requires two devices out of a maximum of 256. Although this allows for a maximum of 128 CLAW connections per interface, a maximum of 32 CLAW connections per interface is recommended.

The CLAW packing feature enables the transport of multiple IP packets in a single channel operation and significantly increases throughput performance between a mainframe and a CMCC adapter. Currently, IBM's TCP/IP stack does not support the CLAW packing feature.

The CLAW packing feature requires changes to the mainframe CLAW driver support. In partnership with Cisco, Interlink Computer Science (now Sterling Software) has made the corresponding CLAW driver change to Cisco IOS for S/390 Release 2 and Interlink TCPaccess 5.2. Customers must make the necessary changes to their host configurations to enable the CLAW packing feature.

For details about configuring a CMCC adapter for CLAW, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

TCP/IP Offload

The Cisco TCP/IP offload feature supports IBM's MVS, VM, and Transaction Processing Facility (TPF) operating systems. The TCP/IP offload feature for CMCC adapters delivers the same function as the TCP/IP offload function on the 3172 Interconnect Controller (Model 3), but with increased performance.

For details about configuring a CMCC adapter for TCP/IP offload, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

IP Host Backup

You can connect multiple mainframes to a single CMCC adapter using an ESCON director. Often, these mainframes run using the ESCON Multiple Image Facility (EMIF), which permits the physical machine to be divided into multiple logical partitions (LPARs). By defining an unused partition on another mainframe, a user can move the operating system from a failed mainframe or mainframe partition to the unused partition. By having multiple paths to each device, the move is accomplished without changing the mainframe software. This function also permits moving an IP stack between multiple operating system images.

On the CMCC adapter, each IP connection is treated as a physical device. The CMCC adapter does not support multiple active paths to a single IP connection (or device). Prior to IP Host Backup, the router configuration had to be changed whenever the mainframe operating system was moved from one mainframe or LPAR to another. The IP Host Backup feature permits the mainframe operating system to be moved from one mainframe to another without requiring a change to the router configuration at the time of the move.

Note IP Host Backup does not provide single system image or automatic failover to a waiting backup application. Host operator action on the mainframe is required in these instances.

For more information about configuring a CMCC adapter for IP host backup, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

Cisco Multipath Channel+

CMPC+ is the Cisco implementation of IBM's MPC+ feature. The CMPC+ feature supports the MPC+ features and protocols necessary to support IP. CMPC+ enables High Performance Data Transfer (HPDT). It allows TCP/IP connections to the host through CMCC adapters, using either the TCP/IP stack or the High Speed Access Services (HSAS) IP stack.

CMPC+ offers the following support:

- Support for TCP/IP and HSAS Transmission Group (TG)
- Support for one IP stack per MPC+ group
- Support for one read subchannel and one write subchannel per CMPC+ group. The read subchannel and write subchannel in an MPC+ group can be on different physical channels.
- Support for up to 64 KB per I/O block
- Runs on the CIP and the CPA

Up to 64 MPC+ groups can be configured on a CMCC, depending on memory configuration.

The CMPC+ feature can coexist with the CLAW, TCP/IP Offload, CSNA, CMPC, and TN3270 server features on the same CMCC adapter.

For details about configuring a CMCC adapter for CMPC+, see the "Configuring CMPC+" chapter in this publication.

CMCC Adapter Features for SNA Environments

The Cisco IOS software supports the following features for CMCC adapters in SNA environments:

Cisco SNA

The CSNA feature provides support for SNA protocols to the IBM mainframe from Cisco 7500, Cisco 7200, and Cisco 7000 with RSP7000 series routers, using CMCC adapters (over both ESCON and parallel interfaces). As an IBM 3172 replacement, a CMCC adapter in a Cisco router supports the External Communications Adapter (XCA) feature of the Virtual Telecommunications Access Method (VTAM).

Support for the XCA feature allows VTAM to define the CMCC's Token Ring devices as switched devices. XCA support also allows the CMCC adapter to provide an alternative to FEPs at sites where the NCP is not required for SNA routing functions.

The CSNA feature supports communication between a channel-attached mainframe and the following types of devices attached to a LAN or WAN:

- PU 2.0 SNA node
- PU 2.1 SNA node
- PU 5/4 SNA node

CSNA also supports communication between two mainframes running VTAM that are either channel-attached to the same CMCC adapter card, or channel-attached to different CMCC adapter cards.

The CSNA feature provides SNA connectivity through a MAC address that is defined on an internal adapter in a CMCC. The internal adapter is a virtual adapter that emulates the LAN adapter in an IBM 3172 Interconnect Controller. Each internal adapter is defined in a corresponding XCA major node in VTAM, which provides an access point (LAN gateway) to VTAM for SNA network nodes.

The internal adapter is configured on an internal (virtual) Token Ring LAN located in the CMCC. Each CMCC can be configured with multiple internal Token Ring LANs and internal adapters. Each internal Token Ring LAN must be configured to participate in source-route bridging to communicate with the LAN devices attached to the router.

By providing Cisco Link Services (CLS) and the LLC2 protocol stack on the CMCC adapter card, all frames destined to or from the CMCC adapter card are switched by the router. The presentation of LAN media types allows the CSNA feature to take advantage of current SRB, RSRB, DLSw+, SR/TLB, internal SDLLC, QLLC services, and APPN functionality through SNASw.

The CSNA feature can coexist with the CLAW, TCP/IP Offload, CMPC, CMPC+, and TN3270 server features on the same CMCC adapter.

For details about configuring a CMCC adapter for CSNA, see the "Configuring CSNA and CMPC" chapter in this publication.

Cisco Multipath Channel

CMPC is Cisco System's implementation of IBM's MultiPath Channel (MPC) feature on Cisco 7500, Cisco 7200, and Cisco 7000 with RSP7000 series routers. CMPC allows VTAM to establish Advanced-Peer-to-Peer Networking (APPN) connections using both High Performance Routing (HPR) and Intermediate Session Routing (ISR) through channel-attached router platforms.

Routers configured for CMPC can be deployed in Parallel MVS Systems Complex (sysplex) configurations.

CMPC can be used to establish an APPN connection between VTAM and the following types of APPN nodes:

- VTAM on another host that is channel-attached to the same CMCC adapter
- VTAM on another host that is channel-attached to a different CMCC adapter in the same router
- TN3270 server using Dependent LU Requester (DLUR) in the same CMCC adapter
- SNASw in the router with the CMCC adapter
- Other APPN nodes external to the CMCC adapter and router such as Communications Server/2, AS/400, other LAN- or WAN-attached VTAM hosts, or remote routers

One read subchannel and one write subchannel are supported for each MPC TG. The read subchannel and write subchannel may be split over two physical channel connections on the same CMCC adapter.

CMPC insulates VTAM from the actual network topology. The MPC protocols are terminated on the CMCC adapter and converted to LLC protocols. After they are converted to LLC protocols, other Cisco features can be used to connect VTAM to other APPN nodes in the network. CMPC can be used in conjunction with DLSw+, RSRB, SR/TLB, SRB, SDLLC, QLLC, ATM LAN emulation, and FRAS host to provide connectivity to VTAM.

CMPC supports connections to PU 2.1 nodes: APPN NN, APPN EN, and LEN. Subarea connections are not supported.

The CMPC feature can coexist with the CLAW, TCP/IP Offload, CSNA, CMPC+, and TN3270 server features on the same CMCC adapter.

For details about configuring a CMCC adapter for CMPC, see the "Configuring CSNA and CMPC" chapter of this guide.

TN3270 Server

TN3270 communications in a TCP/IP network consist of the following basic elements:

- TN3270 client—Emulates a 3270 display device for communication with a mainframe application through a TN3270 server over an IP network. The client can support the standard TN3270 functions (as defined by RFC 1576) or the enhanced functionality provided by TN3270E (defined in RFC 2355). TN3270 clients are available on a variety of operating system platforms.
- TN3270 server—Converts the client TN3270 data stream to SNA 3270 and transfers the data to and from the mainframe.
- Mainframe—Provides the application for the TN3270 client and communicates with the TN3270 server using VTAM.

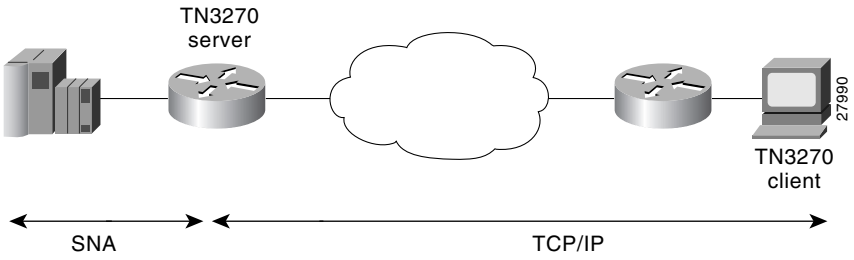
The TN3270 server feature offers an attractive solution when the following conditions need to be supported in an SNA environment:

- Maintaining an IP backbone while providing support for SNA 3270-type clients.
- Offloading mainframe CPU cycles when using a TN3270 host TCP/IP stack with a TN3270 server.
- Providing support for high session density or high transactions per second.

The TN3270 server feature on a CMCC adapter card provides mapping between an SNA 3270 host and a TN3270 client connected to a TCP/IP network as shown in Figure 113. Functionally, it is useful to view the TN3270 server from two different perspectives:

- SNA Functions, page 49
- Telnet Server Functions, page 49

Figure 113 TN3270 Implementation



SNA Functions

From the perspective of an SNA 3270 host connected to the CMCC adapter, the TN3270 server is an SNA device that supports multiple PUs, with each PU supporting up to 255 LUs. The LU can be Type 1, 2, or 3. The SNA host is unaware of the existence of the TCP/IP extension on the implementation of these LUs.

The LUs implemented by the TN3270 server are dependent LUs. To route these dependent LU sessions to multiple VTAM hosts connected to the TN3270 server in the CMCC adapter card, rather than routing in the VTAM hosts, the TN3270 server implements a SNA session switch with EN DLUR function. SNA session switching allows you to eliminate SNA subarea routing between hosts of TN3270 traffic by establishing APPN links with the primary LU hosts directly.

Using the DLUR function is optional so that the TN3270 server can be used with VTAM versions prior to version 4.2, which provide no APPN support. In these non-APPN environments, access to multiple hosts is accomplished using direct PU configuration in the TN3270 server.

Telnet Server Functions

From the perspective of a TN3270 client, the TN3270 server is a high-performance Telnet server that supports Telnet connections, negotiation and data format. The server on the CMCC adapter card supports Telnet connection negotiation and data format as specified in RFC 1576 (referred to as *Traditional TN3270*) and RFC 2355 (referred to as *TN3270 Enhancements*).

Unless the TN3270 server uses a Token Ring connection to a FEP, or other LLC connectivity to the mainframe host, it requires CSNA or CMPC support. For more information about configuring CSNA or CMPC support, see the “Configuring CSNA and CMPC” chapter in this publication.

To enable the TN3270 server feature, you must have a CMCC adapter installed in a Cisco 7000 with RSP7000, Cisco 7500 series router, or a Cisco 7200 router.

For details about configuring the TN3270 server on a CMCC adapter, see the "Configuring the TN3270 Server" chapter in this publication.

Configuring Source-Route Bridging

This chapter describes source-route bridging (SRB) configuration tasks. For a discussion of remote source-route bridging (RSRB) configuration tasks, refer to the "Configuring Remote Source-Route Bridging" chapter in this publication.

For a complete description of the SRB commands mentioned in this chapter, refer to the "Source-Route Bridging Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

Cisco's IOS bridging software includes SRB capability. A source-route bridge connects multiple physical Token Rings into one logical network segment. If the network segment bridges only Token Ring media to provide connectivity, the technology is termed SRB. If the network bridges Token Ring and non-Token Ring media is introduced into the bridged network segment, the technology is termed RSRB.

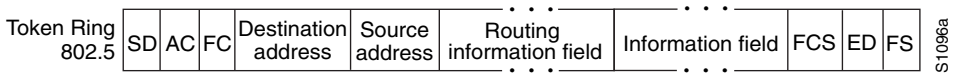
SRB enables routers to simultaneously act as a Level 3 router and a Level 2 source-route bridge. Thus, protocols such as Novell's IPX or XNS can be routed on Token Rings, while other protocols such as Systems Network Architecture (SNA) or NetBIOS are source-route bridged.

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

SRB technology is a combination of bridging and routing functions. A source-route bridge can make routing decisions based on the contents of the MAC frame header. Keeping the routing function at the MAC, or Level 2, layer allows the higher-layer protocols to execute their tasks more efficiently and allows the LAN to be expanded without the knowledge of the higher-layer protocols.

As designed by IBM and the IEEE 802.5 committee, source-route bridges connect extended Token Ring LANs. A source-route bridge uses the RIF in the IEEE 802.5 MAC header of a datagram (Figure 22) to determine which rings or Token Ring network segments the packet must transit.

Figure 22 IEEE 802.5 Token Ring Frame Format



The source station inserts the RIF into the MAC header immediately following the source address field in every frame, giving this style of bridging its name. The destination station reverses the routing field to reach the originating station.

The information in a RIF is derived from explorer packets generated by the source node. These explorer packets traverse the entire source-route bridge network, gathering information on the possible paths the source node might use to send packets to the destination.

Transparent spanning-tree bridging requires time to recompute a topology in the event of a failure; SRB, which maintains multiple paths, allows fast selection of alternate routes in the event of failure. Most importantly, SRB allows the end stations to determine the routes the frames take.

SRB Features

Cisco's SRB implementation has the following features:

- Provides configurable fast-switching software for SRB.
- Provides for a local source-route bridge that connects two or more Token Ring networks.
- Provides *ring groups* to configure a source-route bridge with more than two network interfaces. A ring group is a collection of Token Ring interfaces in one or more routers that are collectively treated as a *virtual ring*.
- Provides two types of explorer packets to collect RIF information—an *all-routes* explorer packet, which follows all possible paths to a destination ring, and a *spanning-tree* explorer packet, which follows a statically configured limited route (spanning tree) when looking for paths.
- Provides a dynamically determined RIF cache based on the protocol. The software also allows you to add entries manually to the RIF cache.
- Provides for filtering by MAC address, link service access point (LSAP) header, and protocol type.
- Provides for filtering of NetBIOS frames either by station name or by a packet byte offset.
- Provides for translation into transparently bridged frames to allow source-route stations to communicate with nonsource-route stations (typically on Ethernet).
- Provides support for the SRB MIB variables as described in the IETF draft "Bridge MIB" document, "Definition of Managed Objects for Bridges," by E. Decker, P. Langille, A. Rijsinghani, and K. McCloghrie, June 1991. Only the SRB component of the Bridge MIB is supported.

- Provides support for the Token Ring MIB variables as described in RFC 1231, *IEEE 802.5 Token Ring MIB*, by K. McCloghrie, R. Fox, and E. Decker, May 1991. Cisco implements the mandatory tables (Interface Table and Statistics Table), but not the optional table (Timer Table) of the Token Ring MIB. The Token Ring MIB has been implemented for the 4/16-Mb Token Ring cards that can be user adjusted for either 4- or 16-Mb transmission speeds (CSC-1R, CSC-2R, CSC-R16M, or CSC-C2CTR).

- SRB is supported over FDDI on Cisco 7200 series routers.
- Particle-based switching is supported (over FDDI and Token Ring) by default on Cisco 7200 series routers.
- Complies with RFC 1483 in Cisco IOS Release 12.0(3)T and later by offering the ability to encapsulate SRB traffic using RFC 1483 bridged LLC encapsulation. This support enables SRB over ATM functionality that is interoperable with other vendors' implementations of SRB over ATM.

SRB Configuration Task List

Perform the tasks in the following sections to configure SRB:

- Configuring Source-Route Bridging, page 4
- Configuring Bridging of Routed Protocols, page 11
- Configuring Translation Between SRB and Transparent Bridging Environments, page 13
- Configuring NetBIOS Support, page 17
- Configuring LNM Support, page 21
- Configuring ATM Support, page 27
- Securing the SRB Network, page 28
- Tuning the SRB Network Task List, page 35
- Establishing SRB Interoperability with Specific Token Ring Implementations, page 39

See the "SRB Configuration Examples" section on page 40 for examples.

Caution The Cisco IOS software issues a warning if a duplicate bridge definition exists in a router. You must remove an old bridge definition before adding a new bridge definition.

Configuring Source-Route Bridging

The Cisco implementation of source-route bridging enables you to connect two or more Token Ring networks using either Token Ring or Fiber Distributed Data Interface (FDDI) media. You can encapsulate source-route bridging traffic over Frame Relay using RFC 1490 Bridged 802.5 encapsulation.

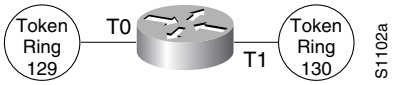
You can configure the Cisco IOS software for source-route bridging by performing the tasks in one of the first three sections and, optionally, the tasks in the last section:

- Configuring a Dual-Port Bridge, page 5
- Configuring a Multiport Bridge Using a Virtual Ring, page 6
- Configuring SRB over FDDI, page 7
- Configuring Fast-Switching SRB over FDDI, page 8
- Configuring SRB over Frame Relay, page 9
- Enabling the Forwarding and Blocking of Spanning-Tree Explorers, page 9
- Enabling the Automatic Spanning-Tree Function, page 10
- Limiting the Maximum SRB Hops, page 11

Configuring a Dual-Port Bridge

A dual-port bridge is the simplest source-route bridging configuration. When configured as a dual-port bridge, the access server or router serves to connect two Token Ring LANs. One LAN is connected through one port (Token Ring interface), and the other LAN is connected through the other port (also a Token Ring interface). Figure 23 shows a dual-port bridge.

Figure 23 Dual-Port Bridge



To configure a dual-port bridge that connects two Token Rings, you must enable source-route bridging on each of the Token Ring interfaces that connect to the two Token Rings. To enable source-route bridging, use the following command in interface configuration mode for each of the Token Ring interfaces:

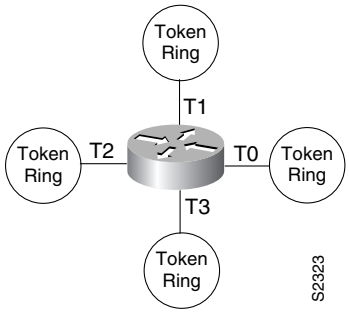
Command	Purpose
`Router(config-if)# source-bridge` *local-ring bridge-number target-ring*	Configures an interface for SRB.

Note Ring numbers need to be unique across interfaces and networks, so that when you enable source-route bridging over an interface the local and target rings are defined. Each node on the network will know if it is the target of explorer packets sent on the network.

A dual-port bridge is a limitation imposed by IBM Token Ring chips; the chips can process only two ring numbers. If you have a router with two or more Token Ring interfaces, you can work around the two-ring number limitation. You can configure your router as multiple dual-port bridges or as a multiport bridge using a virtual ring.

You can define several separate dual-port bridges in the same router. However, the routers on the LANs cannot have any-to-any connectivity; that is, they cannot connect to every other router on the bridged LANs. Only the routers connected to the dual-port bridge can communicate with one another. Figure 24 shows two separate dual-port bridges (T0-T2 and T1-T3) configured on the same router.

Figure 24 Multiple Dual-Port Bridges



To configure multiple dual-port source-route bridges, use the following command in interface configuration mode for each Token Ring interface that is part of a dual-port bridge:

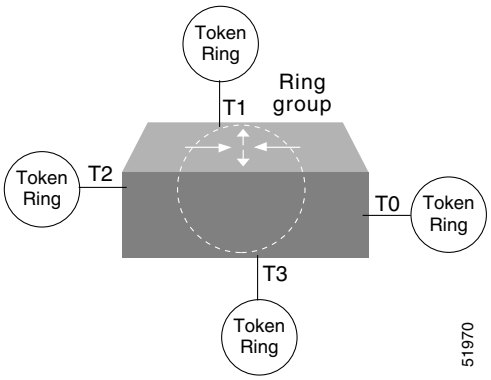
Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Configures an interface for SRB.

If you want your network to use only SRB, you can connect as many routers as you need via Token Rings. Remember, source-route bridging requires you to bridge only Token Ring media.

Configuring a Multiport Bridge Using a Virtual Ring

A better solution for overcoming the two-ring number limitation of IBM Token Ring chips is to configure a multiport bridge using a virtual ring. A virtual ring on a multiport bridge allows the router to interconnect three or more LANs with any-to-any connectivity; that is, connectivity between any of the routers on each of the three LANs is allowed. A virtual ring creates a logical Token Ring internal to the Cisco IOS software, which causes all the Token Rings connected to the router to be treated as if they are all on the same Token Ring. The virtual ring is called a *ring group*. Figure 25 shows a multiport bridge using a virtual ring.

Figure 25 Multiport Bridge Using a Virtual Ring



To take advantage of this virtual ring feature, each Token Ring interface on the router must be configured to belong to the same ring group. For information about configuring a multiport bridge using a virtual ring, see the "Configuring a Multiport Bridge Using a Virtual Ring" section on page 6.

To configure a source-route bridge to have more than two network interfaces, you must perform the following tasks:

1. Define a ring group.
2. Enable source-route-bridging and assign a ring group to a Token Ring interface.

Once you have completed these tasks, the router acts as a multiport bridge, not as a dual-port bridge.

Note Ring numbers need to be unique across interfaces and networks.

Defining a Ring Group in SRB Context

Because all IBM Token Ring chips can process only two ring numbers, we have implemented the concept of a ring group or virtual ring. A ring group is a collection of Token Ring interfaces in one or more routers that share the same ring number. This ring number is used just like a physical ring number, showing up in any route descriptors contained in packets being bridged. Within the context of a multiport bridge that uses SRB rather than RSRB, the ring group resides in the same router. See the "Configuring Remote Source-Route Bridging" chapter to compare ring groups in the SRB and RSRB context.

A ring group must be assigned a ring number that is unique throughout the network. It is possible to assign different Token Ring interfaces on the same router to different ring groups, if, for example, you plan to administer them as interfaces in separate domains.

To define or remove a ring group, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
`Router(config)# no source-bridge ring-group ring-group [virtual-mac-address]`	Removes a ring group.

Enabling SRB and Assigning a Ring Group to an Interface

After you have defined a ring group, you must assign that ring group to those interfaces you plan to include in that ring group. An interface can only be assigned to one ring group. To enable any-to-any connectivity among the end stations connected through this multiport bridge, you must assign the same target ring number to all Token Ring interfaces on the router.

To enable SRB and assign a ring group to an interface, use the following command in interface configuration mode:

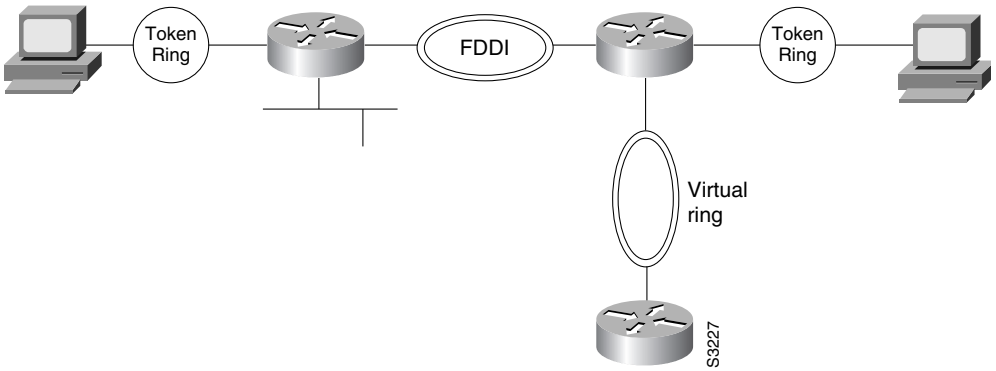
Command	Purpose
`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Configures an interface for SRB.

Configuring SRB over FDDI

Cisco's implementation of SRB expands the basic functionality to allow autonomous switching of SRB network traffic for FDDI interfaces, adding counters to SRB accounting statistics, and implementing process-level switching of SRB over FDDI. This functionality provides a significant increase in performance for Token Rings interconnected across an FDDI backbone (Figure 26).

SRB over FDDI is supported on the Cisco 4000-M, Cisco 4500-M, Cisco 4700-M, Cisco 7000 series, Cisco 7200 series, and Cisco 7500 routers.

Figure 26 Autonomous FDDI SRB



To configure autonomous FDDI SRB, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface fddi slot/port`	Configures an FDDI interface.
Step 2	`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Configures an interface for SRB.
Step 3	`Router(config-if)# source-bridge route-cache cbus`	Enables autonomous switching.

Configuring Fast-Switching SRB over FDDI

Fast-Switching SRB over FDDI enhances performance. For example, if you want to use access-lists, fast-switching SRB over FDDI provides fast performance and access-list filters capability.

To configure fast-switching SRB over FDDI, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface fddi slot/port`	Configures an FDDI interface.
Step 2	`Router(config-if)# source-bridge local-ring bridge-number target-ring`	Configures an interface for SRB.
Step 3	`Router(config-if)# source-bridge spanning`	Enables source-bridge spanning.
Step 4	`Router(config-if)# source-bridge route-cache`	Enables fast-switching.
Step 5	`Router(config-if)# multiring protocol-keyword`	Enables the collection and use of RIF information.

Configuring SRB over Frame Relay

Cisco IOS software offers the ability to encapsulate SRB traffic using RFC 1490 Bridged 802.5 encapsulation. This provides SRB over Frame Relay functionality that is interoperable with other vendors' implementations of SRB over Frame Relay and with some vendors' implementations of FRAS BAN.

Note In this release, SRB over Frame Relay does not support the Cisco IOS software proxy explorer, automatic spanning-tree, or LAN Network Manager functions.

To configure SRB over Frame Relay, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface serial number`	Specifies the serial port.
Step 2	`Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 3	`Router(config-if)# interface serial slot/port.subinterface-number point-to-point`	Configures a Frame Relay point-to-point subinterface.
Step 4	`Router(config-if)# frame-relay interface-dlci dlci ietf`	Configures a DLCI number for the point-to-point subinterface.
Step 5	`Router(config-if)# source-bridge source-ring-number bridge-number target-ring-number conserve-ring`	Assigns a ring number to the Frame Relay permanent virtual circuit.

Enabling the Forwarding and Blocking of Spanning-Tree Explorers

When trying to determine the location of remote destinations on a source-route bridge, the source device will need to send explorer packets. Explorer packets are used to collect routing information field (RIF) information. The source device can send spanning-tree explorers or all-routes explorers. Note that some older IBM devices generate only all-routes explorer packets, but many newer IBM devices are capable of generating spanning-tree explorer packets.

A spanning-tree explorer packet is an explorer packet that is sent to a defined group of nodes that comprise a statically configured spanning tree in the network. In contrast, an all-routes explorer packet is an explorer packet that is sent to every node in the network on every path.

Forwarding all-routes explorer packets is the default. However, in complicated source-route bridging topologies, using this default can generate an exponentially large number of explorers that are traversing the network. The number of explorer packets becomes quite large because duplicate explorer packets are sent across the network to every node on every path. Eventually each explorer packet will reach the destination device. The destination device will respond to each of these explorer packets. It is from these responses that the source device will collect the RIF and determine which route it will use to communicate with the destination device. Usually, the route contained in the first returned response will be used.

The number of explorer packets traversing the network can be reduced by sending spanning-tree explorer packets. Spanning-tree explorer packets are sent to specific nodes; that is, to only the nodes on the spanning tree, not to all nodes in the network. You must manually configure the spanning-tree topology over which the spanning-tree explorers are sent. You do this by configuring which interfaces on the routers will forward spanning-tree explorers and which interfaces will block them.

To enable forwarding of spanning-tree explorers on an outgoing interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge spanning`	Enables the forwarding of spanning-tree explorer packets on an interface.

Note While enabling the forwarding of spanning-tree explorer packets is not an absolute requirement, it is strongly recommended in complex topologies. Configuring an interface to block or forward spanning-tree explorers has no effect on how that interface handles all-routes explorer packets. All-routes explorers can always traverse the network.

To block forwarding of spanning tree explorers on an outgoing interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# no source-bridge spanning`	Blocks spanning-tree explorer packets on an interface.

Enabling the Automatic Spanning-Tree Function

The automatic spanning-tree function supports automatic resolution of spanning trees in SRB networks, which provides a single path for spanning explorer frames to traverse from a given node in the network to another. Spanning explorer frames have a single-route broadcast indicator set in the routing information field. Port identifiers consist of ring numbers and bridge numbers associated with the ports. The spanning-tree algorithm for SRB does not support Topology Change Notification bridge protocol data unit (BDPU).

Note Although the automatic spanning-tree function can be configured with source-route translational bridging (SR/TLB), the SRB domain and transparent bridging domain have separate spanning trees. Each Token Ring interface can belong to only one spanning tree. Only one bridge group can run the automatic spanning-tree function at a time.

To create a bridge group that runs an automatic spanning-tree function compatible with the IBM SRB spanning-tree implementation, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bridge` *bridge-group* `protocol ibm`	Creates a bridge group that runs the automatic spanning-tree function.

To enable the automatic spanning-tree function for a specified group of bridged interfaces, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge spanning` *bridge-group*	Enables the automatic spanning-tree function on a group of bridged interfaces.

To assign a path cost for a specified interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge spanning` *bridge-group* `path-cost` *path-cost*	Assigns a path cost for a specified group of bridged interfaces.

Note Ports running IEEE and IBM protocols form a spanning tree together on the LAN, but they do not mix in the router itself. Make sure the configurations are correct and that each LAN runs only one protocol.

See the end of this chapter for an example of source-route bridging with the automatic spanning-tree function enabled.

Limiting the Maximum SRB Hops

You can minimize explorer storms if you limit the maximum number of source-route bridge hops. For example, if the largest number of hops in the best route between two end stations is six, it might be appropriate to limit the maximum source-route bridging hops to six to eliminate unnecessary traffic. This setting affects spanning-tree explorers and all-routes explorers sent from source devices.

To limit the number of SRB hops, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge max-hops` *count*	Controls the forwarding or blocking of all-routes explorer frames received on this interface.
`Router(config-if)# source-bridge max-in-hops` *count*	Controls the forwarding or blocking of spanning-tree explorer frames received on this interface.
`Router(config-if)# source-bridge max-out-hops` *count*	Controls the forwarding or blocking of spanning-tree explorer frames sent from this interface.

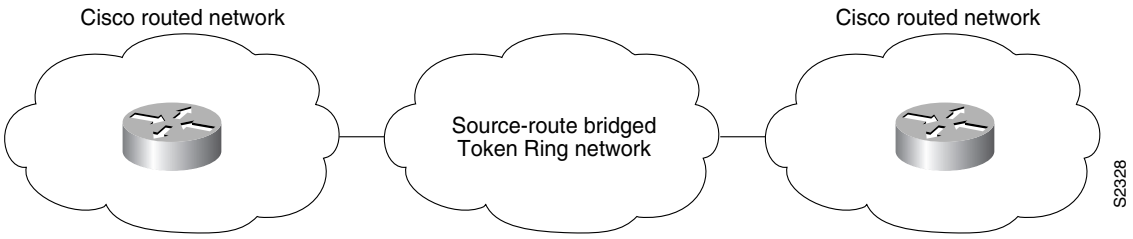
Configuring Bridging of Routed Protocols

Source-route bridges use Media Access Control (MAC) information, specifically the information contained in the RIF, to bridge packets. A RIF contains a series of ring and bridge numbers that represent the possible paths the source node might use to send packets to the destination. Each ring number in the RIF represents a single Token Ring in the source-route bridged network and is designated by a unique 12-bit ring number. Each bridge number represents a bridge that is between two Token Rings in the SRB network and is designated by a unique 4-bit bridge number. The information in a RIF is derived from explorer packets traversing the source-route bridged network. Without the RIF information, a packet could not be bridged across a source-route bridged network.

Unlike source-route bridges, Level 3 routers use protocol-specific information (for example, Novell Internetwork Packet Exchange (IPX) or Xerox Network Systems (XNS) headers) rather than MAC information to route datagrams. As a result, the Cisco IOS software default for routed protocols is to not collect RIF information and to not be able to bridge routed protocols. However, if you want the software to bridge routed protocols across a source-route bridged network, the software must be able to collect

and use RIF information to bridge packets across a source-route bridged network. You can configure the software to append RIF information to routed protocols so that routed protocols can be bridged. Figure 27 shows a network topology in which you would want to use this feature.

Figure 27 Topology for Bridging Routed Protocols across a Source-Route Bridged Network



To configure the Cisco IOS software to bridge routed protocols, perform the following tasks:

- Enabling Use of the RIF, page 12 (Required)
- Configuring a Static RIF Entry, page 13 (Optional)
- Configuring the RIF Timeout Interval, page 13 (Optional)

Enabling Use of the RIF

You can configure the Cisco IOS software so that it will append RIF information to the routed protocols. This allows routed protocols to be bridged across a source-route bridged network. The routed protocols that you can bridge are as follows:

- Apollo Domain
- AppleTalk
- ISO Connectionless Network Service (CLNS)
- DECnet
- IP
- IPX
- VINES
- XNS

Enable use of the RIF only on Token Ring interfaces on the router.

To configure the Cisco IOS software to append RIF information, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# multiring {protocol-keyword [all-routes | spanning] | all | other}`	Enables collection and use of RIF information.

For an example of how to configure the software to bridge routed protocols, see the "SRB and Routing Certain Protocols Example" section on page 43.

Configuring a Static RIF Entry

If a Token Ring host does not support the use of IEEE 802.2 TEST or XID datagrams as explorer packets, you might need to add static information to the RIF cache of the router.

To configure a static RIF entry, use the following command in global configuration mode:

Command	Purpose
`Router(config)# rif mac-address rif-string {interface-name | ring-group ring}`	Enters static source-route information into the RIF cache.

Configuring the RIF Timeout Interval

RIF information that can be used to bridge routed protocols is maintained in a cache whose entries are aged.

Note The **rif validate enable** commands have no effect on remote entries learned over RSRB.

To configure the number of minutes an inactive RIF entry is kept in the cache, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# rif timeout minutes`	Specifies the number of minutes an inactive RIF entry is kept.
Step 2	`Router(config)# rif validate-enable`	Enables RIF validation for entries learned on an interface (Token Ring or FDDI).
Step 3	`Router(config)# rif validate-enable-age`	Enables RIF validation on an SRB that is malfunctioning.
Step 4	`Router(config)# rif validate-enable-route-cache`	Enables synchronization of the RIF cache with the protocol route cache.

Configuring Translation Between SRB and Transparent Bridging Environments

Source-route translational bridging (SR/TLB) is a Cisco IOS software feature that allows you to combine SRB and transparent bridging networks without the need to convert all of your existing source-route bridges to source-route transparent (SRT) nodes. As such, it provides a cost-effective connectivity path between Ethernets and Token Rings, for example.

When a router is configured for SR/TLB, the router operates in fast-switching mode by default, causing packets to be processed in the interrupt handler when the packets first arrive, rather than queueing them for scheduled processing. You can also use the **no source-bridge transparent fastswitch** command to disable fast-switched SR/TLB, causing the router to handle packets by process switching. For more information on disabling fast-switched SR/TLB, refer to the "Disabling Fast-Switched SR/TLB" section on page 16.

Note When you are translationally bridging, you will have to route routed protocols and translationally bridge all others, such as local-area transport (LAT).

Overview of SR/TLB

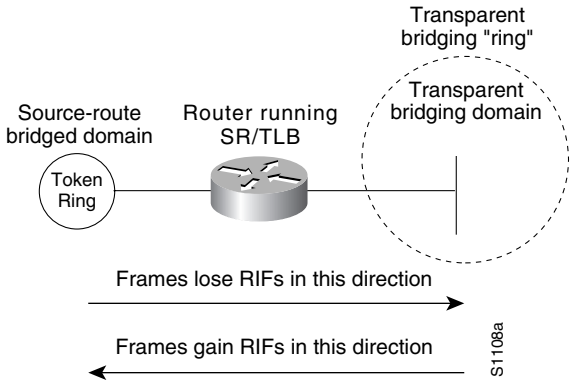
You can bridge packets between an SRB domain and a transparent bridging domain. Using this feature, a software "bridge" is created between a specified virtual ring group and a transparent bridge group. To the source-route station, this bridge looks like a standard source-route bridge. There is a ring number and a bridge number associated with a ring that actually represents the entire transparent bridging domain. To the transparent bridging station, the bridge represents just another port in the bridge group.

When bridging from the SRB (typically, Token Ring) domain to the transparent bridging (typically, Ethernet) domain, the source-route fields of the frames are removed. The RIFs are cached for use by subsequent return traffic.

When bridging from the transparent bridging domain to the SRB domain, the router checks the packet to see if it has a multicast or broadcast destination or a unicast (single host) destination. If it is multicast, the packet is sent as a spanning-tree explorer. If it is a unicast destination, the router looks up the path to the destination in the RIF cache. If a path is found, it will be used; otherwise, the router will send the packet as a spanning-tree explorer.

An example of a simple SR/TLB topology is shown in Figure 28.

Figure 28 Example of a Simple SR/TLB Topology



Note The Spanning Tree Protocol messages used to prevent loops in the transparent bridging domain are *not* passed between the SRB domain and the transparent bridging domain. Therefore, you must not set up multiple paths between the SRB and transparent bridging domains.

The following notes and caveats apply to all uses of SR/TLB:

- Multiple paths cannot exist between the source-route bridged domain and the transparent bridged domain. Such paths can lead to data loops in the network, because the spanning-tree packets used to avoid these loops in transparent bridging networks do not traverse the SRB network.
- Some devices, notably PS/2s under certain configurations running OS/2 Extended Edition Version 1.3, do not correctly implement the "largest frame" processing on RIFs received from remote source-route bridged hosts. The maximum Ethernet frame size is smaller than that allowed for Token Ring. As such, bridges allowing for communication between Ethernet and Token Ring will tell the Token Ring hosts, through the RIF on frames destined to the Token Ring, that hosts on the Ethernet cannot receive frames larger than a specified maximum, typically 1472 bytes. Some machines ignore this run-time limit specification and send frames larger than the Ethernet can accept. The router and any other Token Ring/Ethernet bridge has no choice but to drop these frames. To allow such hosts to successfully communicate across or to an Ethernet, you must configure their maximum frame sizes manually. For the PS/2, this can be done through Communications Manager.

- Any access filters applied on any frames apply to the frames as they appear on the media to which the interface with the access filter applies. This is important because in the most common use of SR/TLB (Ethernet and Token Ring connectivity), the bit ordering of the MAC addresses in the frame is swapped. Refer to the SR/TLB examples in the "SRB Configuration Examples" section of this chapter.

Caution Bridging between dissimilar media presents several problems that can prevent communication from occurring. These problems include bit order translation (or usage of MAC addresses as data), maximum transmission unit (MTU) differences, frame status differences, and multicast address usage. Some or all of these problems might be present in a multimedia bridged LAN and prevent communication from taking place. Because of differences in the way end nodes implement Token Ring, these problems are most prevalent when bridging between Token Rings and Ethernets or between Token Ring and FDDI LANs.

Problems can occur with the following protocols when bridged between Token Ring and other media: Novell IPX, DECnet Phase IV, AppleTalk, VINES, XNS, and IP. Further, problems can occur with the Novell IPX and XNS protocols when bridged between FDDI and other media. Cisco recommends that these protocols be routed whenever possible.

To enable SR/TLB, you must perform the task in the following section:

- Enabling Bridging between Transparent Bridging and SRB, page 15

In addition, you can also perform the tasks in the following sections:

- Disabling Fast-Switched SR/TLB, page 16
- Enabling Translation Compatibility with IBM 8209 Bridges, page 16
- Enabling Token Ring LLC2-to-Ethernet Conversion, page 16

Enabling Bridging between Transparent Bridging and SRB

Before enabling bridging, you must have completely configured your router using multiport SRB and transparent bridging. Once you have done this, to establish bridging between transparent bridging and source-route bridging, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge transparent ring-group pseudo-ring bridge-number tb-group [oui]`	Enables bridging between transparent bridging and SRB.

Disabling Fast-Switched SR/TLB

To disable fast-switched SR/TLB and cause the router to handle packets by process switching, use the following command in global configuration mode:

Command	Purpose
`Router(config)# no source-bridge transparent` *ring-group* **fastswitch**	Disables fast-switched SR/TLB.

Enabling Translation Compatibility with IBM 8209 Bridges

To transfer data between IBM 8209 Ethernet/Token Ring bridges and routers running the SR/TLB software (to create a Token Ring backbone to connect Ethernets), use the following command on each Token Ring interface in interface configuration mode:

Command	Purpose
`Router(config-if)#` **ethernet-transit-oui** [**90-compatible** | **standard** | **cisco**]	Moves data between IBM 8209 Ethernet/Token Ring bridges and routers running translational bridging software.

Enabling Token Ring LLC2-to-Ethernet Conversion

The Cisco IOS software supports the following types of Token Ring-to-Ethernet frame conversions using Logical Link Control, type 2 (LLC2) Protocol:

- Token Ring LLC2 to Ethernet Type II (0x80d5 processing)
- Token Ring LLC2 to Ethernet 802.3 LLC2 (standard)

For most non-IBM hosts, Token Ring LLC2 frames can be translated in a straightforward manner into Ethernet 802.3 LLC2 frames. This is the default conversion in the Cisco IOS software.

However, many Ethernet-attached IBM devices use nonstandard encapsulation of LLC2 on Ethernet. Such IBM devices, including PS/2s running OS/2 Extended Edition and RT-PCs, do not place their LLC2 data inside an 802.3 format frame, but rather place it into an Ethernet Type 2 frame whose type is specified as *0x80d5*. This nonstandard format is called 0x80d5, named after the type of frame. This format is also sometimes called *RT-PC Ethernet format* because these frames were first widely seen on the RT-PC. Hosts using this nonstandard 0x80d5 format cannot read the standard Token Ring LLC2 to Ethernet 802.2 LLC frames.

To enable Token Ring LLC2 to Ethernet LLC2 conversion, you can perform one or both of the following tasks:

- Enable 0x80d5 Processing, page 16
- Enable Standard Token Ring LLC2-to-Ethernet LLC2 Conversion, page 17

Enable 0x80d5 Processing

You can change the Cisco IOS software's default translation behavior of translating Token Ring LLC to Ethernet 802.3 LLC to translate Token Ring LLC2 frames into Ethernet 0x80d5 format frames. To enable this nonstandard conversion, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **source-bridge enable-80d5**	Changes the Ethernet/Token Ring translation behavior to translate Token Ring LLC2 frames into Ethernet 0x80d5 format frames.

Enable Standard Token Ring LLC2-to-Ethernet LLC2 Conversion

After you change the translation behavior to perform Token Ring LLC2 frames into Ethernet 0x80d5 format frames, some of the non-IBM hosts in your network topology might use the standard Token Ring conversion of Token Ring LLC2 to 802.3 LLC2 frames. If this is the case, you can change the translation method of those hosts to use the standard translation method on a per-DSAP basis. The translation method for all the IBM hosts would still remain as Token Ring LLC2 to Ethernet 0x80d5 translation.

To define non-IBM hosts in your network topology to use the standard translation method while the IBM hosts use the nonstandard method, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge sap-80d5 dsap`	Allows some other devices to use normal LLC2/IEEE 802.3 translation on a per-DSAP basis.

Configuring NetBIOS Support

NetBIOS is a nonroutable protocol that was originally designed to send messages between stations, typically IBM PCs, on a Token Ring network. NetBIOS allows messages to be exchanged between the stations using a name rather than a station address. Each station knows its name and is responsible for knowing the names of other stations on the network.

Note In addition to this type of NetBIOS, which runs over LLC2, we have implemented another type of NetBIOS that runs over IPX. For information on the IPX type of NetBIOS, refer to the chapter "Configuring Novell IPX" in the *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

NetBIOS name caching allows the Cisco IOS software to maintain a cache of NetBIOS names, which avoids the high overhead of sending many of the broadcasts used between client and server NetBIOS PCs (IBM PCs or PS/2s) in an SRB environment.

When NetBIOS name caching is enabled, the software performs the following actions:

- Notices when any hosts send a series of duplicated "query" frames and reduces them to one frame per period. The time period is configurable.
- Keeps a cache of mappings between NetBIOS server and client names and their MAC addresses. By watching NAME_QUERY and NAME_RECOGNIZED request and response traffic between clients and servers, the Cisco IOS software can forward broadcast requests sent by clients to find servers (and by servers in reply to their clients) directly to their needed destinations, rather than forwarding them for broadcast across the entire bridged network.

The software will time out the entries in the NetBIOS name cache after a specific interval of their initial storage. The timeout value is a user-configurable value. You can configure the timeout value for a particular Token Ring if the NetBIOS name cache is enabled on the interface connecting to that Token Ring. In addition, you can configure static name cache entries that never time out for frequently accessed servers whose locations or paths typically do not change. Static RIF entries are also specified for such hosts.

Generally, NetBIOS name caching is most useful when a large amount of NetBIOS broadcast traffic creates bottlenecks on WAN media connecting distant locations, and the WAN media is overwhelmed with this traffic. However, when two high-speed LAN segments are directly interconnected, the packet savings of NetBIOS name caching is probably not worth the processor overhead associated with it.

Note NetBIOS name caching is not recommended to be turned on in backbone routers, particularly if you have it enabled in all the routers connected to the backbone. NetBIOS caching should be distributed among multiple routers. NetBIOS name caching can be used only between Cisco routers that are running software Release 9.1 or later.

To enable NetBIOS name caching, you must perform the tasks in the following sections:

- Enabling the Proxy Explorers Feature on the Appropriate Interface, page 18
- Specifying Timeout and Enabling NetBIOS Name Caching, page 19

In addition, you can configure NetBIOS name caching as described in the following sections:

- Configuring the NetBIOS Cache Name Length, page 19
- Enabling NetBIOS Proxying, page 19
- Creating Static Entries in the NetBIOS Name Cache, page 20
- Specifying Dead-Time Intervals for NetBIOS Packets, page 20

Enabling the Proxy Explorers Feature on the Appropriate Interface

To enable NetBIOS name caching on an interface, the proxy explorers feature must first be enabled on that interface. This feature must either be enabled for response to all explorer packets or for response to NetBIOS packets only.

To determine whether the proxy explorers feature has been enabled, use the following command in privileged EXEC mode:

Command	Purpose
`Router# more nvram:startup-config`	Displays the startup configuration file contained in NVRAM or specified by the CONFIG_FILE environment variable.

To determine whether proxy explorers has been configured for response to all explorer packets, look in the configuration file for the **source-bridge proxy-explorer** entry for the appropriate interface. For example, if the appropriate interface is Token Ring 0, look for an entry similar to the following:

```
interface tokenring 0
source-bridge proxy-explorer
```

If that entry does not exist, look for the **source-bridge proxy-netbios-only** entry for the appropriate interface.

If neither entry exists, proxy explorers has not yet been enabled for the appropriate interface. To enable proxy explorers for response to all explorer packets, refer to the section “Configure Proxy Explorers” later in this chapter.

Otherwise, enable proxy explorers only for the NetBIOS name caching function by using the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge proxy-netbios-only`	Enables use of proxy explorers only for the NetBIOS name caching function and not for their general local response to explorers.

Specifying Timeout and Enabling NetBIOS Name Caching

After you have ensured that the proxy explorers feature has been enabled for the appropriate interface, you can specify a cache timeout and enable NetBIOS name caching. To do this, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# netbios name-cache timeout minutes`	Enables NetBIOS name caching and sets the time that entries can remain in the NetBIOS name cache.
Step 2	`Router(config)# netbios enable-name-cache`	Enables NetBIOS name caching.

Configuring the NetBIOS Cache Name Length

To specify how many characters of the NetBIOS type name that the name cache will validate, use the following command in global configuration mode:

Command	Purpose
`Router(config)# netbios name-cache name-len length`	Specifies how many characters of the NetBIOS type name the name cache will validate.

Enabling NetBIOS Proxying

The Cisco IOS software can act as a proxy and send NetBIOS datagram type frames. To enable this capability, use the following command in global configuration mode:

Command	Purpose
`Router(config)# netbios name-cache proxy-datagram seconds`	Enables the Cisco IOS software to act as a proxy and send NetBIOS datagram type frames.

To define the validation time when the software is acting as a proxy for NetBIOS NAME_QUERY command or for explorer frames, use the following global configuration command:

Command	Purpose
`Router(config)# rif validate-age seconds`	Defines validation time.

Creating Static Entries in the NetBIOS Name Cache

If the router communicates with one or more NetBIOS stations on a regular basis, adding static entries to the NetBIOS name cache for these stations can reduce network traffic and overhead. You can define a static NetBIOS name cache entry that associates the server with the NetBIOS name and the MAC address. If the router acts as a NetBIOS server, you can specify that the static NetBIOS name cache is available locally through a particular interface. If a remote router acts as the NetBIOS server, you can specify that the NetBIOS name cache is available remotely. To do this, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# netbios name-cache mac-address netbios-name interface-name`	Defines a static NetBIOS name cache entry, tying the server with the name netbios-name to the mac-address, and specifying that the server is accessible locally through the interface-name specified.
`Router(config)# netbios name-cache mac-address netbios-name ring-group group-number`	Defines a static NetBIOS name cache entry, tying the server with the name netbios-name to the mac-address, and specifying that the server is accessible remotely through the ring-group group-number specified.

If you have defined a NetBIOS name cache entry, you must also define a RIF entry. For an example of how to configure a static NetBIOS entry, see the "NetBIOS Support with a Static NetBIOS Cache Entry Example" section on page 51.

Specifying Dead-Time Intervals for NetBIOS Packets

When NetBIOS name caching is enabled and default parameters are set on the router (and the NetBIOS name server and the NetBIOS name client), approximately 20 broadcast packets per login are kept on the local ring where they are generated. The broadcast packets are of the type ADD_NAME_QUERY, ADD_GROUP_NAME, and STATUS_QUERY.

The Cisco IOS software also converts pairs of FIND_NAME and NAME_RECOGNIZED packets received from explorers, which traverse all rings, to specific route frames that are sent only between the two machines that need to see these packets.

You can specify a query-timeout, or "dead-time" interval to prevent repeat or duplicate broadcast of these type of packets for the duration of the interval.

To specify dead time intervals, use the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# netbios name-cache query-timeout seconds`	Specifies a dead time interval during which the Cisco IOS software drops any broadcast (NetBIOS ADD_NAME_QUERY, ADD_GROUP_NAME, or STATUS_QUERY) frames if they are duplicate frames sent by the same host.
`Router(config)# netbios name-cache recognized-timeout seconds`	Specifies a dead time interval during which the software drops FIND_NAME and NAME_RECOGNIZED frames if they are duplicate frames sent by the same host.

Configuring LNM Support

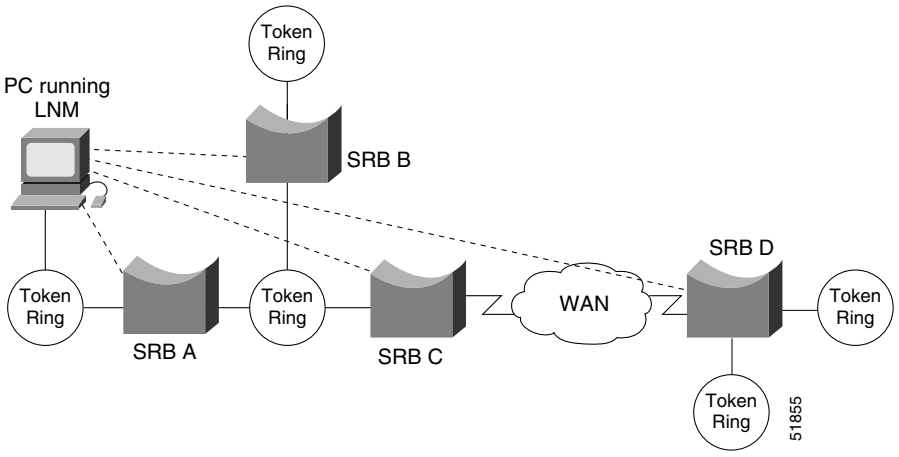
LAN Network Manager (LNM), formerly called LAN Manager, is an IBM product for managing a collection of source-route bridges. Using either a proprietary protocol or the Simple Network Management Protocol (SNMP), LNM allows you to monitor the entire collection of Token Rings that comprise your source-route bridged network. You can use LNM to manage the configuration of source-route bridges, monitor Token Ring errors, and gather information from Token Ring parameter servers.

Note LNM is supported on the 4/16-Mb Token Ring cards that can be configured for either 4- or 16-Mb transmission speeds. LNM support is not provided on CSC-R16M cards with SBEMON 2.0.

LNM is not limited to managing locally attached Token Ring networks; it also can manage any other Token Rings in your source-route bridged network that are connected through non-Token Ring media. To accomplish this task, LNM works in conjunction with the IBM Bridge Program. The IBM Bridge Program gathers data about the local Token Ring network and relays it back to LNM. In this manner, the bridge program becomes a proxy for information about its local Token Ring. Without this ability, you would require direct access to a device on every Token Ring in the network. This process would make managing an SRB environment awkward and cumbersome.

Figure 29 shows some Token Rings attached through a cloud and one LNM linking to a source-route bridge on each local ring.

Figure 29 LNM Linking to a Source-Route Bridge on Each Local Ring



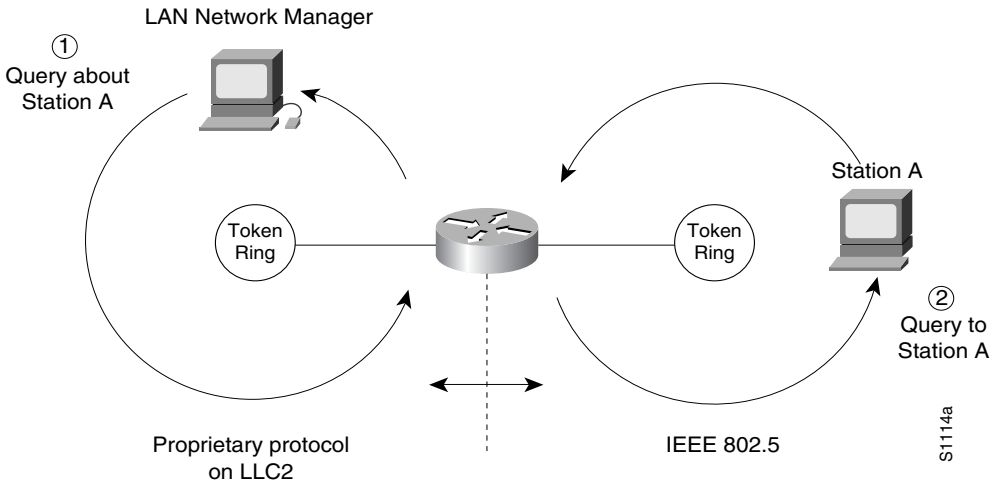
If LNM requires information about a station somewhere on a Token Ring, it uses a proprietary IBM protocol to query to one of the source-route bridges connected to that ring. If the bridge can provide the requested information, it simply responds directly to LNM. If the bridge does not have the necessary information, it queries the station using a protocol published in the IEEE 802.5 specification. In either case, the bridge uses the proprietary protocol to send a valid response back to LNM, using the proprietary protocol.

As an analogy, consider a language translator who sits between a French-speaking diplomat and a German-speaking diplomat. If the French diplomat asks the translator a question in French for the German diplomat and the translator knows the answer, he or she simply responds without translating the original question into German. If the French diplomat asks a question the translator does not know how to answer, the translator must first translate the question to German, wait for the German diplomat to answer, and then translate the answer back to French.

Similarly, if LNM queries a source-route bridge in the proprietary protocol and the bridge knows the answer, it responds directly using the same protocol. If the bridge does not know the answer, it must first translate the question to the IEEE 802.5 protocol, query the station on the ring, and then translate the response back to the proprietary protocol to send to LNM.

Figure 30 illustrates requests from the LNM originating in an IBM proprietary protocol and then translated into IEEE 802.5 MAC-level frames.

Figure 30 LAN Network Manager Monitoring and Translating



Notice that the proprietary protocol LNM uses to communicate with the source-route bridge is an LLC2 connection. Although its protocol cannot be routed, LNM can monitor or manage anything within the SRB network.

How a Router Works with LNM

Cisco routers using 4/16-Mbps Token Ring interfaces configured for SRB support the proprietary protocol that LNM uses. These routers provide all functions the IBM Bridge Program currently provides. Thus LNM can communicate with a router as if it were an IBM source-route bridge, such as the IBM 8209, and can manage or monitor any Token Ring connected to the router.

Through IBM Bridge support, LNM provides three basic services for the SRB network:

- The Configuration Report Server (CRS) monitors the current logical configuration of a Token Ring and reports any changes to LNM. CRS also reports various other events, such as the change of an active monitor on a Token Ring.
- The Ring Error Monitor (REM) monitors errors reported by any station on the ring. In addition, REM monitors whether the ring is in a functional or a failure state.
- The Ring Parameter Server (RPS) reports to LNM when any new station joins a Token Ring and ensures that all stations on a ring are using a consistent set of reporting parameters.

IBM Bridge support for LNM also allows asynchronous notification of some events that can occur on a Token Ring. Examples of these events include notification of a new station joining the Token Ring or of the ring entering failure mode, known as *beaconing*. Support is also provided for LNM to change the operating parameters in the bridge. For a complete description of LNM, refer to the IBM product manual supplied with the LNM program.

LNM support in our source-route bridges is a powerful tool for managing SRB networks. Through the ability to communicate with LNM and to provide the functionality of the IBM Bridge Program, our device appears as part of the IBM network. You therefore gain from the interconnectivity of our products without having to learn a new management product or interface.

When SRB is enabled on the router, configuring the Cisco IOS software to perform the functions of an IBM Bridge for communication with LNM occurs automatically. Therefore, if SRB has been enabled, you do not need to perform any tasks to enable LNM support. However, the LNM software residing on a management station on a Token Ring on the network should be configured to properly communicate with the router.

There are several options for modifying LNM parameters in the Cisco IOS software, but none are required for basic functionality. For example, because users can now modify the operation of the Cisco IOS software through SNMP and through LNM, there is an option to exclude a user from modifying the Cisco IOS software configuration through LNM. You also can specify which of the three LNM services (CRS, REM, RPS) the source-route bridge will perform.

To configure LNM support, perform the tasks in the following sections:

Configuring LNM Software on the Management Stations to Communicate with the Router

Because configuring an LNM station is a fairly simple task and is well covered in the LNM documentation, it is not covered in depth here. However, it is important to mention that you must enter the MAC addresses of the interfaces comprising the ports of the bridges as adapter addresses. When you configure the router as a multiport bridge, configuring an LNM station is complicated by the virtual ring that is involved. The basic problem extends from the fact that LNM is designed to only understand the concept of a two-port bridge, and the router with a virtual ring is a *multiport* bridge. The solution is to configure a virtual ring into the LNM Manager station as a series of dual-port bridges.

Disabling LNM Functionality

Under some circumstances, you can disable all LNM server functions on the router without having to determine whether to disable a specific server, such as the ring parameter server or the ring error monitor on a given interface.

To disable LNM functionality, use the following command in global configuration mode:

Command	Purpose
`Router(config)# lnm disabled`	Disables LNM functionality.

The command can be used to terminate all LNM server input and reporting links. In normal circumstances, this command should not be necessary because it is a superset of the functions normally performed on individual interfaces by the **no lnm rem** and **no lnm rps** commands.

Disabling Automatic Report Path Trace Function

Under some circumstances, such as when new hardware has been introduced into the network and is causing problems, the automatic report path trace function can be disabled. The new hardware may be setting bit-fields B1 or B2 (or both) of the routing control field in the routing information field embedded in a source-route bridged frame. This condition may cause the network to be flooded by report path trace frames if the condition is persistent. The **lnm pathtrace-disabled** command, along with its options, allows you to alleviate network congestion that may be occurring by disabling all or part of the automatic report path trace function within LNM.

To disable the automatic report path trace function, use the following command in global configuration mode:

Command	Purpose
`Router(config)# lnm pathtrace-disabled [all | origin]`	Disables LNM automatic report path trace function.

Preventing LNM Stations from Modifying Cisco IOS Software Parameters

Because there is more than one way to remotely change parameters in a router (either using SNMP or the proprietary IBM protocol), some method is needed to prevent such changes from detrimentally interacting with each other.You can prevent any LNM station from modifying parameters in the Cisco IOS software. It does not affect the ability of LNM to monitor events, only to change parameters on the router.

To prevent the modification of Cisco IOS software parameters by an LNM station, use the following command in global configuration mode:

Command	Purpose
`Router(config)# lnm snmp-only`	Prevents LNM stations from modifying LNM parameters in the Cisco IOS software.

Enabling Other LRMs to Change Router Parameters

LNM has a concept of reporting links and reporting link numbers. A reporting link is simply a connection (or potential connection) between a LAN Reporting Manager (LRM) and a bridge. A reporting link number is a unique number used to identify a reporting link. An IBM bridge allows four simultaneous reporting links numbered 0 through 3. Only the LRM attached on the lowest-numbered connection is allowed to change LNM parameters in the router, and then only when that connection number falls below a certain configurable number. In the default configuration, the LRM connected through link 0 is the only LRM that can change LNM parameters.

To enable other LRMs to change router parameters, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lnm alternate number`	Enables a LRM other than that connected through link 0 to change router parameters.

Applying a Password to an LNM Reporting Link

Each reporting link has its own password that is used not only to prevent unauthorized access from an LRM to a bridge but to control access to the different reporting links. This is important because it is possible to change parameters through some reporting links.

To apply a password to an LNM reporting link, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lnm password number string`	Applies a password to an LNM reporting link.

Enabling LNM Servers

As in an IBM bridge, the router provides several functions that gather information from a local Token Ring. All of these functions are enabled by default, but also can be disabled. The LNM servers are explained in the "How a Router Works with LNM" section on page 22.

To enable LNM servers, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# lnm crs`	Enables the LNM Configuration Report Server (CRS).
`Router(config-if)# lnm rem`	Enables the LNM Ring Error Monitor (REM).
`Router(config-if)# lnm rps`	Enables the LNM Ring Parameter Server (RPS).

Changing Reporting Thresholds

The Cisco IOS software sends a message to all attached LNMs whenever it begins to drop frames. The threshold at which this report is generated is based on a percentage of frames dropped compared with those forwarded. This threshold is configurable, and defaults to a value of 0.10 percent. You can configure the threshold by entering a single number, expressing the percentage loss rate in hundredths of a percent. The valid range is 0 to 9999.

To change reporting thresholds, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lnm loss-threshold number`	Changes the threshold at which the Cisco IOS software reports the frames-lost percentage to LNM.

Changing an LNM Reporting Interval

All stations on a Token Ring notify the Ring Error Monitor (REM) when they detect errors on the ring. In order to prevent excessive messages, error reports are not sent immediately, but are accumulated for a short interval and then reported. A station learns the duration of this interval from a router (configured as a source-route bridge) when it first enters the ring. This value is expressed in tens of milliseconds between error messages. The default is 200, or 2 seconds. The valid range is 0 to 65535.

To change an LNM reporting interval, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lnm softerr milliseconds`	Sets the time interval in which the Cisco IOS software will accumulate error messages before sending them.

Enabling the RPS Express Buffer Function

The RPS express buffer function allows the router to set the express buffer bit to ensure priority service for frames required for ring station initiation. When this function is enabled, the router sets the express buffer bit in its initialize ring station response. This allows Token Ring devices to insert into the ring during bursty conditions.

To enable LNM to use the RPS express buffer function, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lnm express-buffer`	Enables the LNM RPS express buffer function.

Monitoring LNM Operation

Once LNM support is enabled, you can monitor LNM operation. To observe the configuration of the LNM bridge and its operating parameters, use the following commands in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# show lnm bridge`	Displays all configured bridges and their global parameters.
Step 2	`Router# show lnm config`	Displays the logical configuration of all bridges configured in the router.
Step 3	`Router# show lnm interface [type number]`	Displays LNM information for an interface or all interfaces of the router.
Step 4	`Router# show lnm ring [ring-number]`	Displays LNM information about a Token Ring or all Token Rings on the network.
Step 5	`Router# show lnm station [address]`	Displays LNM information about a station or all known stations on all rings.

Configuring ATM Support

Cisco IOS software supports RFC 1483, enabling the transfer of network interconnect traffic over ATM AAL5 layer using LLC encapsulation. RFC 1483 defines an encapsulation type for transferring LAN data via ATM networks. All LAN protocols that use the LLC format and run on Ethernet, Token Ring, or ATM networks are encapsulated in LLC data packets transported via ATM networks. This enhancement provides an SRB over ATM functionality that is interoperable with other vendors' implementations of SRB over ATM.

RFC 1483 also provides the following benefits:

- Flexibility to implement traffic policies pertaining to traffic shaping and various congestion control mechanisms
- Load balancing of traffic guarantees that LAN data is sent
- Cost effectiveness of using PVCs instead of LANE in small networks
- Transfer of connectionless LAN data over a connection-oriented ATM network
- Support for IP and IPX routing, using RFC 1483 Routed PDUs

RFC 1483 enables SRB between Token Ring LANs connected over and ATM network, using RFC 1483 bridged PDUs in the following scenarios:

  - Two-port and multiport SRB between Token Ring LANs connected via RFC 1483 AAL5Snap permanent virtual circuits (PVCs), using bridged PDUs.
  - Two-port and multiport SRB between Token Ring LANs (using RFC 1483 AAL5Snap PVCs) and LANs, VLANs, or ELANs with SRB (using bridged PDUs).

RFC 1483 also supports two-port and multiport Source Route/Translational Bridging (SR/TLB) between Token Ring, Ethernet and their respective emulated LANS, using RFC 1483 bridged PDUs.

SR/TLB can be configured to connect transparent bridging and SRB domains. Transparent bridging forwards incoming packets based on a destination MAC address that yields a RIF to be added to the packet. SRB forwards packets based on destination MAC address, which is listed in the transparent bridging table. Both SRB explorers and transparent bridging multicast packets are forwarded and extended.

The following guidelines apply to RFC 1483 configuration:

- Assign a unique number to the PVC that connects two nodes. When SRB is configured, the router determines the PVC on which the frame is to be forwarded and treats it as a Token Ring interface. In a large network, the availability of enough unique virtual ring numbers for PVCs might be a limitation.
- Conserve the virtual ring number on the PVC and configure the routers so that they use the same ring numbers that are assigned to the PVCs.

To configure SRB over ATM, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface atm slot/port`	Specifies the ATM interface.
Step 2	`Router(config-if)# interface atm slot/port [subinterface- number {multipoint | point-to-point}]`	Specifies the ATM main interface or subinterface to which discovered PVCs will be assigned.

	Command	Purpose
Step 3	`Router(config-if)# atm pvc vcd vpi vci aal-encap [[midlow midhigh] [peak average [burst]]] [inarp [minutes]] [oam [seconds]]`	Creates a PVC on an ATM interface.
Step 4	`Router(config-if)# source-bridge local-ring bridge-number target-ring-number conserve-ring`	Assigns a ring number to the ATM PVC.
Step 5	`Router(config-if)# source-bridge spanning bridge-group`	Enables the automatic spanning-tree function on a group of bridged interfaces.

For more information, see one of the following sections:

- Back-to-Back Routers ATM Configuration Example, page 60
- Single ATM PVC and Single Virtual Ring Per Router Configuration Example, page 61
- Multiple ATM PVCs and Multiple Virtual Rings on One Router Configuration Example, page 62
- Multiple ATM PVCs with a Single Virtual Ring on the Router Configuration Example, page 63

Securing the SRB Network

This section describes how to configure three features that are used primarily to provide network security: NetBIOS access filters, administrative filters, and access expressions that can be combined with administrative filters. In addition, these features can be used to increase network performance because they reduce the number of packets that traverse the backbone network.

Configuring NetBIOS Access Filters

NetBIOS packets can be filtered when sent across a Token Ring bridge. Two types of filters can be configured:

- Host access list

  Used for source and destination station names

- Byte offset access list

  Used for arbitrary byte patterns in the packet itself.

As you configure NetBIOS access filters, keep the following issues in mind:

- The access lists that apply filters to an interface are scanned in the order they are entered.
- There is no way to put a new access list entry in the middle of an access list. All new additions to existing NetBIOS access lists are placed at the end of the existing list.
- Access list arguments are case sensitive. The software makes a literal translation, so that a lowercase “a” is different from an uppercase “A.” (Most nodes are named in uppercase letters.)
- A host NetBIOS access list and byte NetBIOS access list can each use the same name. The two lists are identified as unique and bear no relationship to each other.

- The station names included in the access lists are compared with the source name field for NetBIOS commands 00 and 01 (ADD_GROUP_NAME_QUERY and ADD_NAME_QUERY), and with the destination name field for NetBIOS commands 08, 0A, and 0E (DATAGRAM, NAME_QUERY, and NAME_RECOGNIZED).
- If an access list does not contain a particular station name, the default action is to deny the access to that station.

To minimize any performance degradation, NetBIOS access filters do not examine all packets. Rather, they examine certain packets that are used to establish and maintain NetBIOS client/server connections, thereby effectively stopping new access and load across the router. However, applying a new access filter does not terminate existing sessions immediately. All new sessions will be filtered, but existing sessions could continue for some time.

There are two ways you can configure NetBIOS access filters:

- Configure NetBIOS Access Filters Using Station Names, page 29
- Configuring NetBIOS Access Filters Using a Byte Offset, page 29

Configure NetBIOS Access Filters Using Station Names

To configure access filters using station names, you must do the following:

1. Assign the station access list name.
2. Specify the direction of the message to be filtered on the interface.

The NetBIOS station access list contains the station name to match, along with a permit or deny condition. You must assign the name of the access list to a station or set of stations on the network.

To assign a station access list name, use the following command in global configuration mode:

Command	Purpose
`Router(config)# netbios access-list host` *name* {**permit** | **deny**} *pattern*	Assigns the name of an access list to a station or set of stations on the network.

When filtering by station name, you can choose to filter either incoming or outgoing messages on the interface. To specify the direction, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# netbios input-access-filter host` *name*	Defines an access list filter for incoming messages.
`Router(config-if)# netbios output-access-filter host` *name*	Defines an access list filter for outgoing messages.

Configuring NetBIOS Access Filters Using a Byte Offset

To configure access filters you must do the following:

1. Assign a byte offset access list name.
2. Specify the direction of the message to be filtered on the interface.

Keep the following notes in mind while configuring access filters using a byte offset:

- When an access list entry has an offset plus the length of the pattern that is larger than the packet's length, the entry will not make a match for that packet.
- Because these access lists allow arbitrary byte offsets into packets, these access filters can have a significant impact on the amount of packets per second transiting across the bridge. They should be used only when situations absolutely dictate their use.

The NetBIOS byte offset access list contains a series of offsets and hexadecimal patterns with which to match byte offsets in NetBIOS packets. To assign a byte offset access list name, use the following command in global configuration mode:

Command	Purpose
`Router(config)# netbios access-list bytes` *name* {**permit** | **deny**} *offset pattern*	Defines the byte offsets and patterns within NetBIOS messages to match with access list parameters.

Note Using NetBIOS Byte Offset access filters disables the autonomous or fast switching of source-route bridging frames.

When filtering by byte offset, you can filter either incoming or outgoing messages on the interface. To specify the direction, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# netbios input-access-filter bytes` *name*	Specifies a byte-based access filter on incoming messages.
`Router(config-if)# netbios output-access-filter bytes` *name*	Specifies a byte-based access filter on outgoing messages.

Configuring Administrative Filters for Token Ring Traffic

Source-route bridges normally filter frames according to the routing information contained in the frame. That is, a bridge will not forward a frame back to its originating network segment or any other network segment that the frame has already traversed. This section describes how to configure another type of filter—the administrative filter.

Administrative filters can filter frames based on the following methods:

- Protocol type—IEEE 802 or Subnetwork Access Protocol (SNAP)
- Token Ring vendor code
- Source address
- Destination address

Whereas filtering by Token Ring address or vendor code causes no significant performance penalty, filtering by protocol type significantly affects performance. A list of SNAP (Ethernet) type codes is provided in the "Ethernet Type Codes" appendix in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Filtering Frames by Protocol Type

You can configure administrative filters by protocol type by specifying protocol type codes in an access list. You then apply that access list to either IEEE 802.2 encapsulated packets or to SNAP-encapsulated packets on the appropriate interface.

The order in which you specify these elements affects the order in which the access conditions are checked. Each condition is tested in succession. A matching condition is then used to execute a permit or deny decision. If no conditions match, a deny decision is reached.

Note If a single condition is to be denied, there must be an **access-list** command that permits everything as well, or all access is denied.

To filter frames by protocol type, use the following command in global configuration mode:

Command	Purpose
`Router(config)# access-list` *access-list-number* {**permit** | **deny**} {*type-code wild-mask* | *address mask*}	Creates an access list for filtering frames by protocol type.

You can filter IEEE 802-encapsulated packets on either input or output. The access list you specify is the one you created that includes the protocol type codes.

To enable filtering on input or output, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge input-lsap-list` *access-list-number*	Enables filtering of IEEE 802-encapsulated packets on input by type code.
`Router(config-if)# source-bridge output-lsap-list` *access-list-number*	Enables filtering of IEEE 802-encapsulated packets on output by type code.

You can filter SNAP-encapsulated packets on either input or output. The access list you specify is the one you created that includes the protocol type codes.

To enable filtering on input or output, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge input-type-list` *access-list-number*	Filters SNAP-encapsulated packets on input by type code.
`Router(config-if)# source-bridge output-type-list` *access-list-number*	Filters SNAP-encapsulated frames on output by type code.

Filtering Frames by Vendor Code

To configure administrative filters by vendor code or address, define access lists that look for Token Ring addresses or for particular vendor codes for administrative filtering. To do so, use the following command in global configuration mode:

Purpose	Command
`Router(config)# access-list` *access-list-number* {**permit** | **deny**} *address mask*	Configures vendor code access lists.

Filtering Source Addresses

To configure filtering on IEEE 802 source addresses, assign an access list to a particular input interface for filtering the Token Ring or IEEE 802 source addresses. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **source-bridge input-address-list** *access-list-number*	Enables filtering on IEEE 802 source addresses.

Filtering Destination Addresses

To configure filtering on IEEE 802 destination addresses, assign an access list to a particular output interface. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **source-bridge output-address-list** *access-list-number*	Enables filtering on IEEE 802 destination addresses.

Configuring Access Expressions that Combine Administrative Filters

You can use access expressions to combine access filters to establish complex conditions under which bridged frames can enter or leave an interface. Using access expressions, you can achieve levels of control on the forwarding of frames that otherwise would be impossible when using only simple access filters. Access expressions are constructed from individual access lists that define administrative filters for the following fields in packets:

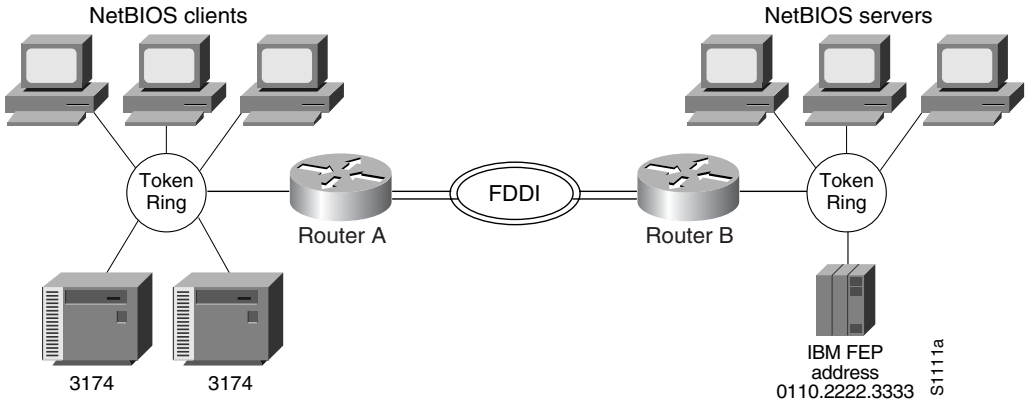
- LSAP and SNAP type codes
- MAC addresses
- NetBIOS station names
- NetBIOS arbitrary byte values

Note For any given interface, an access expression cannot be used if an access list has been defined for a given direction. For example, if an input access list is defined for MAC addresses on an interface, no access expression can be specified for the input side of that interface.

In Figure 31, two routers each connect a Token Ring to an FDDI backbone. On both Token Rings, SNA and NetBIOS bridging support is required. On Token Ring A, NetBIOS clients must communicate with any NetBIOS server off Token Ring B or any other, unpictured router. However, the two 3174 cluster controllers off Token Ring A must only communicate with the one FEP off of Token Ring B, located at MAC address 0110.2222.3333.

Without access expressions, this scenario cannot be achieved. A filter on Router A that restricted access to only the FEP would also restrict access of the NetBIOS clients to the FEP. What is needed is an access *expression* that would state "If it is a NetBIOS frame, pass through, but if it is an SNA frame, only allow access to address 0110.2222.3333."

Figure 31 Access Expression Example



Note Using access-expressions that combine access filters disables the autonomous or fast switching of source-route bridging frames.

Configuring Access Expressions

To configure an access expression perform the following tasks:

- Design the access expression.
- Configure the access lists used by the expression.
- Configure the access expression into the router.

When designing an access expression, you must create some phrase that indicates, in its entirety, all the frames that will *pass* the access expression. This access expression is designed to apply on frames coming from the Token Ring interface on Router A in Figure 31:

> "Pass the frame if it is a NetBIOS frame or if it is an SNA frame destined to address 0110.2222.3333."

In Boolean form, this phrase can be written as follows:

> "Pass if 'NetBIOS or (SNA and destined to 0110.2222.3333).'"

The preceding statement requires three access lists to be configured:

- An access list that passes a frame if it is a NetBIOS frame (SAP = 0xF0F0)
- An access list that passes a frame if it is an SNA frame (SAP = 0x0404)
- An access list that passes a MAC address of 0110.2222.3333

The following configuration allows for all these conditions:

```
! Access list 201 passes NetBIOS frames (command or response)
access-list 201 permit 0xF0F0 0x0001
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
```

The 0x0001 mask allows command and response frames to pass equally.

To apply the access expression to the appropriate interface, enter the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# access-expression {in | out} expression`	Defines a per-interface access expression.

Optimizing Access Expressions

It is possible to combine access expressions. Suppose you wanted to send SNA traffic through to a single address, but allow other traffic through the router without restriction. The phrase could be written as follows:

"Allow access if the frame is not an SNA frame, or if it is going to host 0110.2222.3333."

More tersely, this would be:

"Not SNA or destined to 0110.2222.3333."

The access lists defined in the previous section create the following configuration:

```
interface tokenring 0
 access-expression in ~lsap(202) | dmac(701)
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
 access-list 701 permit 0110.2222.3333
```

This is a better and simpler access list than the one originally introduced and will probably result in better run-time execution as a result. Therefore, it is best to simplify your access expressions as much as possible before configuring them into the Cisco IOS software.

Note An "access-expression" type filter cannot exist with a "source-bridge" type filter on the same interface. The two types of filters are mutually exclusive.

Altering Access Lists Used in Access Expressions

Because access expressions are composed of access lists, special care must be taken when deleting and adding access lists that are referenced in these access expressions.

If an access list that is referenced in an access expression is deleted, the access expression merely ignores the deleted access list. However, if you want to redefine an access list, you can create a new access list with the appropriate definition and use the same name as the old access list. The newly defined access list replaces the old one of the same name.

For example, if you want to redefine the NetBIOS access list named MIS that was used in the preceding example, you would use the following sequence of configuration commands:

```
! Replace the NetBIOS access list
interface tokenring 0
 access-expression in (smac(701) & netbios-host(accept))
 no netbios access-list host accept permit CISCO*
```

Tuning the SRB Network Task List

The following sections describe how to configure features that enhance network performance by reducing the number of packets that traverse the backbone network:

Note In some situations, you might discover that default settings for LLC2 configurations are not acceptable. In such a case, you can configure LLC2 for optimal use. The chapter "Configuring LLC2 and SDLC Parameters" in this publication describes how you can use them to optimize your network performance.

Enabling or Disabling the Source-Route Fast-Switching Cache

Rather than processing packets at the process level, the fast-switching feature enables the Cisco IOS software to process packets at the interrupt level. Each packet is transferred from the input interface to the output interface without copying the entire packet to main system memory. Fast switching allows for faster implementations of local SRB between 4/16-MB Token Ring cards in the same router, or between two routers using the 4/16-Mb Token Ring cards and direct encapsulation.

By default, fast-switching software is enabled when SRB is enabled. To enable or disable source-route fast-switching, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge route-cache`	Enables fast-switching.
`Router(config-if)# no source-bridge route-cache`	Disables fast-switching.

Note Using either NetBIOS Byte Offset access filters or access expressions that combine access filters disables the fast switching of source-route bridging frames.

Enabling or Disabling the Source-Route Autonomous-Switching Cache

Autonomous switching is a feature that enables the Cisco IOS software to send packets from the input ciscoBus card to the output ciscoBus card without any involvement on the part of the router processor.

Autonomous switching is available for local SRB between ciscoBus Token Ring (CTR) cards in the same router. Autonomous switching provides higher switching rates than does fast switching between 4/16-Mb Token Ring cards. Autonomous switching works for both two-port bridges and multiport bridges that use ciscoBus Token Ring cards.

In a virtual ring that includes both ciscoBus Token Ring and 4/16-Mb Token Ring interfaces, frames that flow from one CTR interface to another are autonomously switched, and the remainder of the frames are fast switched. The switching that occurs on the CTR interface takes advantage of the high-speed ciscoBus controller processor.

To enable or disable source-route autonomous switching, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge route-cache cbus`	Enables autonomous switching.
`Router(config-if)# no source-bridge route-cache cbus`	Disables autonomous switching.

Note Using either NetBIOS Byte Offset access filters or access-expressions that combine access filters disables the autonomous switching of SRB frames.

Enabling or Disabling the SSE

The Silicon Switch Engine (SSE) acts as a programmable cache to speed the switching of packets. To enable or disable the SSE, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# source-bridge route-cache sse`	Enables the SSE function.
`Router(config-if)# no source-bridge route-cache sse`	Disables the SSE function.

Establishing the Connection Timeout Interval

It might be necessary to adjust timeout intervals in a complex topology such as a large multihop WAN with virtual rings or satellite links. The timeout interval is used when a connection to a remote peer is attempted. If the timeout interval expires before a response is received, the connection attempt is aborted.

To set the connection timeout interval, use the following command in global configuration mode:

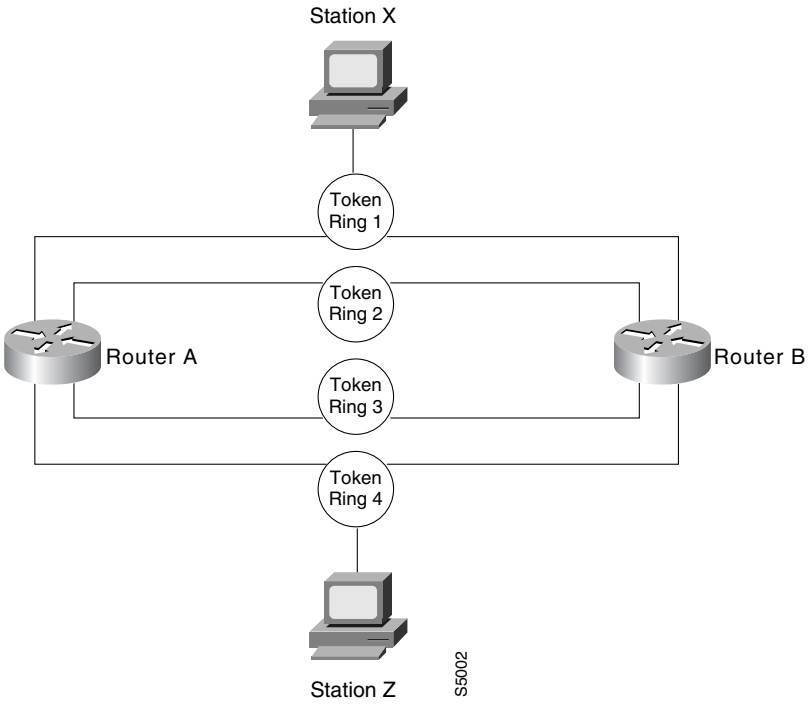
Command	Purpose
`Router(config)# source-bridge connection-timeout` ***seconds***	Sets the connection timeout interval.

Optimizing Explorer Processing

Efficient explorer processing is vital to the operation of SRB. The default configuration is satisfactory for most situations. However, there might be circumstances that create unexpected broadcast storms. You can optimize the handling of explorer frames, thus reducing processor overhead and increasing explorer packet throughput. Optimizing explorer processing enables the router to perform substantially better during explorer broadcast storms.

In networks with redundant topologies—two or more routers connected to the same set of Token Rings and doing source-route bridging—a station on one Token Ring trying to get to a station on another Token Ring may choose a less than optimal route through unnecessary routers, causing explorer storms due to excessive forwarding of explorer frames. For example, in the redundant topology example shown in Figure 32, if Station X on Token Ring 1 attempts to get to Station Z on Token Ring 4 by going through Router A, Token Ring 2, and Router B—a less than optimal route, excessive forwarding of explorer frames may cause explorer storms.

Figure 32 Controlling Explorer Storms in Redundant Network Topologies



The **source-bridge explorer-dup-ARE-filter** command can be used to reduce explorer traffic by filtering explorer frames.

To optimize explorer processing, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# source-bridge explorerq-depth` *depth*	Sets the maximum explorer queue depth.
`Router(config)# source-bridge explorer-dup-ARE-filter`	Prevents explorer storms in redundant network topologies by filtering explorers that have already been forwarded once.
`Router(config)# source-bridge explorer-maxrate` *maxrate*	Sets the maximum byte rate of explorers per ring.

You must also disable explorer fast-switching which is, by default, enabled. To disable explorer fast-switching, use the following command in global configuration mode:

:

Command	Purpose
`Router(config)# no source-bridge explorer-fastswitch`	Disables explorer fast switching.

To enable explorer fast-switching after it has been disabled, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge explorer-fastswitch`	Enables explorer fast switching.

Configuring Proxy Explorers

You can use the proxy explorers feature to limit the amount of explorer traffic propagating through the source-bridge network.

To configure proxy explorers, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# source-bridge proxy-explorer`	Enables the interface to respond to any explorer packets that meet certain conditions necessary for a proxy response to occur.

The Cisco IOS software does not propagate proxy responses for a station. Instead, the software obtains the RIF path from the RIF cache, changes the explorer to a specific frame, and forwards this frame to the destination. If a response is not received before the validation timer expires, the RIF entry is marked as invalid. The invalid RIF entry is flushed from the cache table when another explorer for this station is received, and an explorer is forwarded to discover a path to this station.

Establishing SRB Interoperability with Specific Token Ring Implementations

This section describes how you can establish interoperability between routers and specific Token Ring implementations. It includes the following sections:

Establishing SRB Interoperability with TI MAC Firmware

You can use a workaround to establish interoperability with Texas Instruments MAC firmware.

There is a known defect in earlier versions of the Texas Instruments Token Ring MAC firmware. This implementation is used by Proteon, Apollo, and IBM RTs. A host using a MAC address whose first two bytes are zeros (such as a Cisco router) will not properly communicate with hosts using that version of Texas Instruments firmware.

There are two solutions. The first involves installing a static RIF entry for every faulty node with which the router communicates. If there are many such nodes on the ring, this may not be practical.

You also can set the MAC address of our Token Ring to a value that works around the problem. Resetting the MAC address forces the use of a different MAC address on the specified interface, thereby avoiding the TI MAC firmware problem. However, you must ensure that no other host on the network is using that MAC address.

To reset the MAC address, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# mac-address ieee-address`	Resets the MAC address of the Token Ring interface to a value that provides a workaround to a problem in Texas Instruments Token Ring MAC firmware.

Reporting Spurious Frame-Copied Errors

An IBM 3174 cluster controller can be configured to report frame-copied errors to IBM LAN Network Manager software. These errors indicate that another host is responding to the MAC address of the 3174 cluster controller. Both the 3174 cluster controller and the IBM LAN Network Manager software can be configured to ignore frame-copied errors.

Monitoring and Maintaining the SRB Network

You can display a variety of information about the SRB network. To display the information you require, use one of the following commands in EXEC mode, as needed:

Command	Purpose
`Router# show access-expression [begin | exclude | include]`	Displays the defined input and output access list expressions.
`Router# show controllers token`	Displays internal state information about the Token Ring interfaces in the system.
`Router# show interfaces tokenring`	Provides high-level statistics for a particular interface.

Command	Purpose
`Router# show interfaces`	Provides high-level statistics about the state of source bridging for a particular interface.
`Router# show lnm bridge`	Displays all currently configured bridges and all parameters that are related to the bridge as a whole and not to one of its interfaces.
`Router# show lnm config`	Displays the logical (multiport bridge) configuration of the Cisco IOS software.
`Router# show lnm interface [type number]`	Displays all LNM-relevant information about a specific interface.
`Router# show lnm ring [ring-number]`	Displays all LNM-relevant information about a specific ring number.
`Router# show lnm station [address]`	Displays all LNM-relevant information about a specific station or about all known stations on the ring.
`Router# show local-ack`	Shows the current state of any current local acknowledgment for both LLC2 and SDLLC connections.
`Router# show netbios-cache`	Displays the contents of the NetBIOS cache.
`Router# show rif`	Displays the contents of the RIF cache.
`Router(config)# show source-bridge [interface]`	Displays the current source bridge configuration and miscellaneous statistics.
`Router# show span`	Displays the spanning-tree topology for the router.
`Router# show sse summary`	Displays a summary of Silicon Switch Processor (SSP) statistics.

To maintain the SRB network, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# clear netbios-cache`	Clears the entries of all dynamically learned NetBIOS names.
`Router# clear rif-cache`	Clears the entire RIF cache.
`Router# clear source-bridge`	Clears the SRB statistical counters.
`Router# clear sse`	Reinitializes the SSP on the Cisco 7000 series.

In addition to the EXEC-mode commands to maintain the SRB network, you can use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge tcp-queue-max number`	Limits the size of the backup queue for RSRB to control the number of packets that can wait for transmission to a remote ring before they are thrown away.

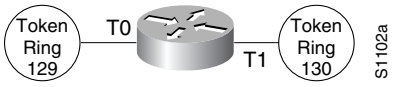
SRB Configuration Examples

The following sections provide SRB configuration examples:

Basic SRB with Spanning-Tree Explorers Example

Figure 33 illustrates a simple two-port bridge configuration. Token Rings 129 and 130 are connected through the router.

Figure 33 Dual-Port Source-Route Bridge Configuration



The example that follows routes IP, but source-route bridges all other protocols using spanning-tree explorers:

```
interface tokenring 0
 ip address 131.108.129.2 255.255.255.0
 source-bridge 129 1 130
```

```
 source-bridge spanning
 multiring all
!
interface tokenring 1
 ip address 131.108.130.2 255.255.255.0
 source-bridge 130 1 129
 source-bridge spanning
! use RIFs, as necessary, with IP routing software
 multiring all
```

SRB with Automatic Spanning-Tree Function Configuration Example

The following example of a Cisco series 7000 router configuration illustrates how to enable the automatic spanning-tree function on an SRB network:

```
source-bridge ring-group 100

interface tokenring 0/0
 no ip address
 ring-speed 16
 multiring all
 source-bridge active 1 10 100
 source-bridge spanning 1
!
interface tokenring 0/1
 no ip address
 ring-speed 16
 multiring all
 source-bridge active 2 10 100
 source-bridge spanning 1
!
bridge 1 protocol ibm
```

Optimized Explorer Processing Configuration Example

The following configuration example improves the handling of explorer frames, enabling the Cisco IOS software to perform substantially better during explorer broadcast storms. In this configuration, the maximum byte rate of explorers is set to 100000.

```
source-bridge explorer-maxrate 100000
source-bridge explorerQ-depth 100
no source-bridge explorer-fastswitch
```

SRB-Only Example

The following example shows that all protocols are bridged, including IP. Because IP is being bridged, the system has only one IP address.

```
no ip routing
!
interface tokenring 0
 ip address 131.108.129.2 255.255.255.0
 source-bridge 129 1 130
 source-bridge spanning
!
interface tokenring 1
 ip address 131.108.129.2 255.255.255.0
```

```
 source-bridge 130 1 129
 source-bridge spanning
!
interface ethernet 0
 ip address 131.108.129.2 255.255.255.0
```

SRB and Routing Certain Protocols Example

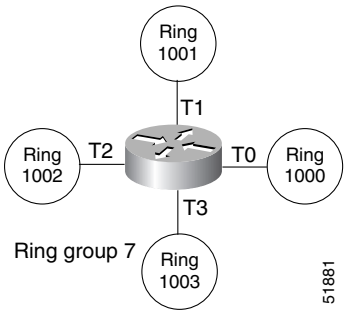
In the following configuration, IP, XNS, and IPX are routed, while all other protocols are bridged between rings. While not strictly necessary, the Novell IPX and XNS network numbers are set consistently with the IP subnetwork numbers. This makes the network easier to maintain.

```
xns routing 0000.0C00.02C3
!
novell routing 0000.0C00.02C3
!
interface tokenring 0
 ip address 131.108.129.2 255.255.255.0
 xns network 129
 novell network 129
 source-bridge 129 1 130
 source-bridge spanning
 multiring all
!
interface tokenring 1
 ip address 131.108.130.2 255.255.255.0
 xns network 130
 novell network 130
 source-bridge 130 1 129
 source-bridge spanning
 multiring all
!
interface ethernet 0
 ip address 131.108.2.68 255.255.255.0
 xns network 2
 novell network 2
```

Multiport SRB Example

Figure 34 shows an example configuration of a four-port Token Ring source-route bridge. Rings 1000, 1001, 1002, and 1003 are all source-route bridged to each other across ring group 7.

Figure 34 Four-Port Source-Route Bridge



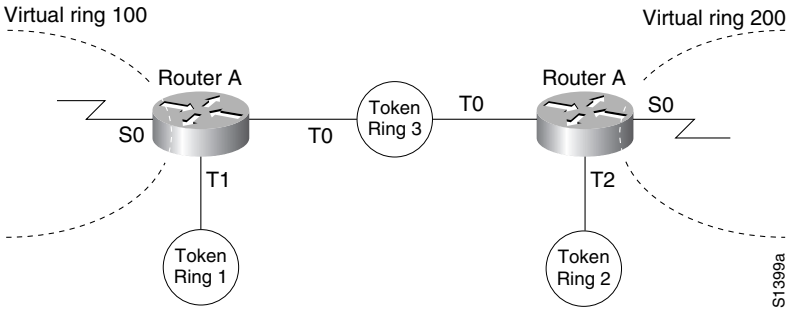
The following is a sample configuration file:

```
source-bridge ring-group 7
!
interface tokenring 0
 source-bridge 1000 1 7
 source-bridge spanning
!
interface tokenring 1
 source-bridge 1001 1 7
 source-bridge spanning
!
interface tokenring 2
 source-bridge 1002 1 7
 source-bridge spanning
!
interface tokenring 3
 source-bridge 1003 1 7
 source-bridge spanning
```

SRB with Multiple Virtual Ring Groups Example

Two virtual ring groups can only be connected through an actual Token Ring. Figure 35 shows virtual rings 100 and 200 connected through Token Ring 3.

Figure 35 Two Virtual Rings Connected by an Actual Token Ring



Configuration for Router A

```
source-bridge ring-group 100
!
interface tokenring 0
 source-bridge 3 4 100
 source-bridge spanning
!
interface tokenring 1
 source-bridge 1 4 100
 source-bridge spanning
```

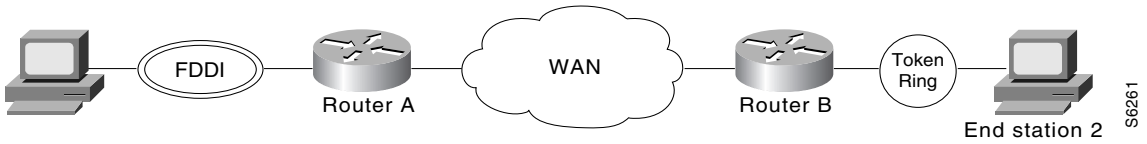
Configuration for Router B

```
source-bridge ring-group 200
!
interface tokenring 0
 source-bridge 3 1 200
 source-bridge spanning
!
interface tokenring 2
 source-bridge 2 1 200
 source-bridge spanning
```

SRB over FDDI Configuration Examples

The following examples show the configuration for SRB over FDDI as illustrated in Figure 36.

Figure 36 SRB over FDDI Configuration



Router A

```
dlsw local-peer peer-id 132.11.11.2
dlsw remote-peer 0 tcp 132.11.11.3
interface Fddi0
 no ip address
 multiring all
 source-bridge 26 1 10
 source-bridge spanning
```

Router B

```
dlsw local-peer peer-id 132.11.11.2
dlsw remote-peer 0 tcp 132.11.11.3
interface TokenRing0
 no ip address
 ring-speed 16
 multiring all
 source-bridge 25 1 10
 source-bridge spanning
```

SRB over FDDI Fast-Switching Example

The following example shows SRB over FDDI fast-switching:

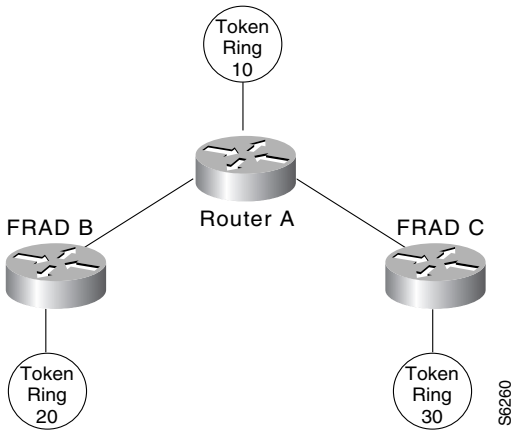
```
interface fddi 2/0
 source-bridge 1 10 2
 source-bridge spanning
 source-bridge route-cache
 multiring ip
```

SRB over Frame Relay Configuration Example

Figure 37 illustrates a network with the following characteristics:

- Virtual Ring Number of Router A = 100
- Virtual Ring Number of FRAD B = 200
- Virtual Ring Number of FRAD C = 300
- DLCI number for PVC between Router A and FRAD B = 30
- DLCI number for PVC between Router A and FRAD C = 31

Figure 37 FRAD Using SRB over Frame Relay to Connect to a Cisco Router



In this example, we configure a new option, **conserve-ring**, on the **source-bridge** interface configuration command. When this option is configured, the SRB software does not add the ring number associated with the Frame Relay PVC (the partner's virtual ring) to outbound explorer frames. This option is permitted for Frame Relay subinterfaces only.

This approach does not require a separate ring number per DLCI. The router configures the partner FRAD's virtual ring number as the ring number for the PVC. FRAD B configures its virtual ring as 200 and the ring for the PVC as 100. FRAD C configures its virtual ring as 300 and the ring for the PVC as 100.

Configuration of Router A

```
source-bridge ring-group 100
!
interface Serial1
 encapsulation frame-relay
!
interface Serial1.1 point-to-point
 frame-relay interface-dlci 30 ietf
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning
!
interface Serial1.2 point-to-point
 frame-relay interface-dlci 31 ietf
 source-bridge 300 1 100 conserve-ring
 source-bridge spanning
!
interface TokenRing0
 source-bridge 500 1 100
```

Configuration on Router B

```
source-bridge ring-group 200
!
interface Serial0
 encapsulation frame-relay
!
interface Serial0.30 point-to-point
 frame-relay interface-dlci 30 ietf
 source-bridge 100 1 200 conserve-ring
 source-bridge spanning
!
interface TokenRing0
source-bridge 600 1 200
```

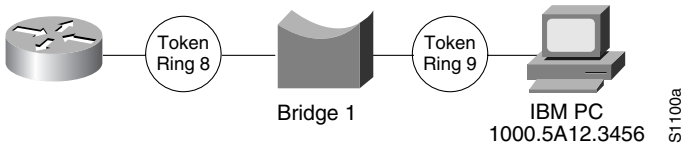
Configuration on Router C

```
source-bridge ring-group 300
!
interface Serial0
 encapsulation frame-relay
!
interface Serial0.31 point-to-point
 frame-relay interface-dlci 31 ietf
 source-bridge 100 1 300 conserve-ring
 source-bridge spanning
!
interface TokenRing0
source-bridge 900 1 300
```

Adding a Static RIF Cache Entry Example

In the example configuration in Figure 38, the path between rings 8 and 9 connected via Bridge 1 is described by the route descriptor 0081.0090. The full RIF, including the route control field, is 0630.0081.0090.

Figure 38 Assigning a RIF to a Source-Route Bridge



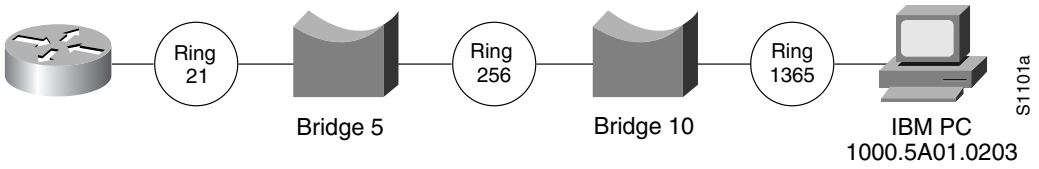
The static RIF entry would be submitted to the router on the left as follows:

```
rif 1000.5A12.3456 0630.0081.0090
```

Adding a Static RIF Cache Entry for a Two-Hop Path Example

In Figure 39, assume that a datagram was sent from a router on ring 21 (15 hexadecimal) across Bridge 5 to ring 256 (100 hexadecimal), then across Bridge 10 (A hexadecimal) to ring 1365 (555 hexadecimal) for delivery to a destination host on that ring.

Figure 39 Assigning a RIF to a Two-Hop Path



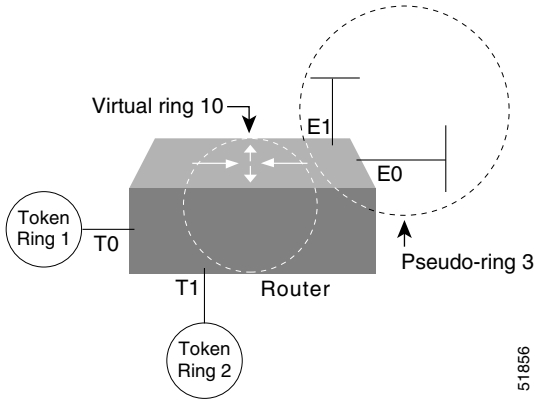
The RIF in the router on the left describing this two-hop path is 0830.0155.100a.5550 and is entered as follows:

```
rif 1000.5A01.0203 0830.0155.100a.5550
```

SR/TLB for a Simple Network Example

In the simple example illustrated in Figure 40, a four-port router with two Ethernets and two Token Rings is used to connect transparent bridging on the Ethernets to SRB on the Token Rings.

Figure 40 Example of a Simple SR/TLB Configuration



Assume that the following configuration for SRB and transparent bridging existed before you wanted to enable SR/TLB:

```
interface tokenring 0
 source-bridge 1 1 2
!
interface tokenring 1
 source-bridge 2 1 1
!
interface ethernet 0
 bridge-group 1
!
interface ethernet 0
 bridge-group 1
!
bridge 1 protocol dec
```

To enable SR/TLB, one aspect of this configuration must change immediately—a third ring must be configured. Before SR/TLB, the two Token Ring interfaces were communicating with two-port local source-route bridging; after SR/TLB, these two interfaces must be reconfigured to communicate through a virtual ring, as follows:

```
source-bridge ring-group 10
!
interface tokenring 0
 source-bridge 1 1 10
!
interface tokenring 1
 source-bridge 2 1 10
!
interface ethernet 0
 bridge-group 1
!
interface ethernet 1
 bridge-group 1
!
bridge 1 protocol dec
```

Now you are ready to determine two things:

- A ring number for the pseudo-ring that is unique throughout the source-route bridged network. For the preceding example configuration, use the number 3.
- A bridge number for the path to the pseudo-ring. For the preceding example configuration, use the number 1.

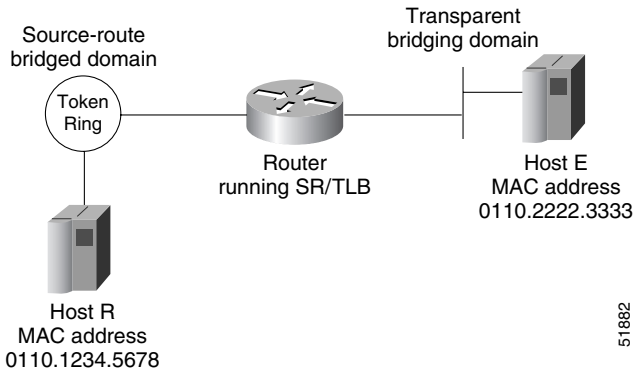
Once you have determined the ring number and the bridge number, you can add the **source-bridge transparent** command to the file, including these two values as parameters for the command. The following partial configuration includes this **source-bridge transparent** entry:

```
source-bridge ring-group 10
source-bridge transparent 10 3 1 1
!
interface tokenring 0
 source-bridge 1 1 10
!
interface tokenring 1
 source-bridge 2 1 10
!
interface ethernet 0
 bridge-group 1
!
interface ethernet 1
 bridge-group 1
!
bridge 1 protocol dec
```

SR/TLB with Access Filtering Example

In the example shown in Figure 41, you want to connect only a single machine, Host E, on an Ethernet to a single machine, Host R, on the Token Ring.

Figure 41 Example of a Bit-Swapped Address



You want to allow only these two machines to communicate across the router. Therefore, you might create the following configuration to restrict the access. However, this configuration will not work, as explained in the paragraph following the sample configuration file.

Note For readability, the commands that control bridging are not shown here, just the commands that control the filtering.

```
interface tokenring 0
 access-expression output smac(701)
!
interface ethernet 0
 bridge-group 1 input-address-list 701
!
 access-list 701 permit 0110.2222.3333
```

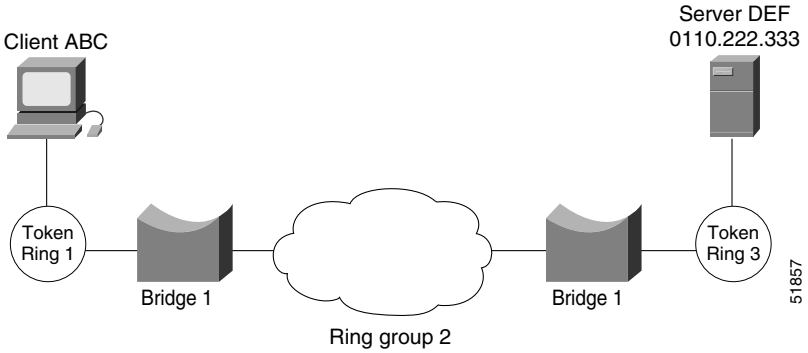
The command for the Token Ring interface specifies that the access list 701 be applied on the source address of frames going out to the Token Ring, and the command for the Ethernet interface specifies that this access list be applied on the source address frames entering the interface from Ethernet. This would work if both interfaces used the same bit ordering, but Token Rings and Ethernets use opposite (swapped) bit orderings in their addresses in relationship to each other. Therefore, the address of Host E on the Token Ring is not 0110.2222.3333, but rather 8008.4444.cccc, resulting in the following configuration. The following configuration is better. This example shows that access lists for Token Ring and Ethernet should be kept completely separate from each other.

```
interface tokenring 0
 source-bridge input-address-list 702
!
interface ethernet 0
 bridge-group 1 input-address-list 701
!
 access-list 701 permit 0110.2222.3333
!
 access-list 702 permit 0110.1234.5678
```

NetBIOS Support with a Static NetBIOS Cache Entry Example

Figure 42 shows a NetBIOS client on a Token Ring connected through a cloud to a NetBIOS server on another Token Ring.

Figure 42 Specifying a Static Entry



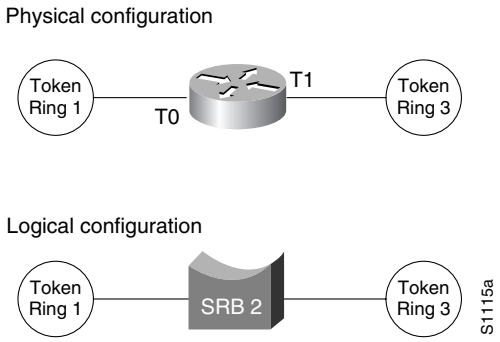
In Figure 42, a static entry is created in the router attached to ring 1 on the client side of the ring group. The static entry is to the server DEF, which is reached through the router attached to ring 3. If server DEF has the MAC address 0110.2222.3333, the configuration for the static entry on the client side is as follows:

```
rif 0110.2222.3333 0630.0021.0030 ring-group 2
netbios name-cache 0110.2222.3333 DEF ring-group 2
```

LNM for a Simple Network Example

Figure 43 shows a router with two Token Rings configured as a local source-route bridge.

Figure 43 Router with Two Token Rings Configured as a Local Source-Route Bridge



The associated configuration file follows:

```
interface tokenring 0
 source-bridge 1 2 3
!
interface tokenring 1
 source-bridge 3 2 1
```

The **show lnm config** command displays the logical configuration of this bridge, including the LNM configuration information that needs to be entered at the LNM Station. A sample **show lnm config** display follows:

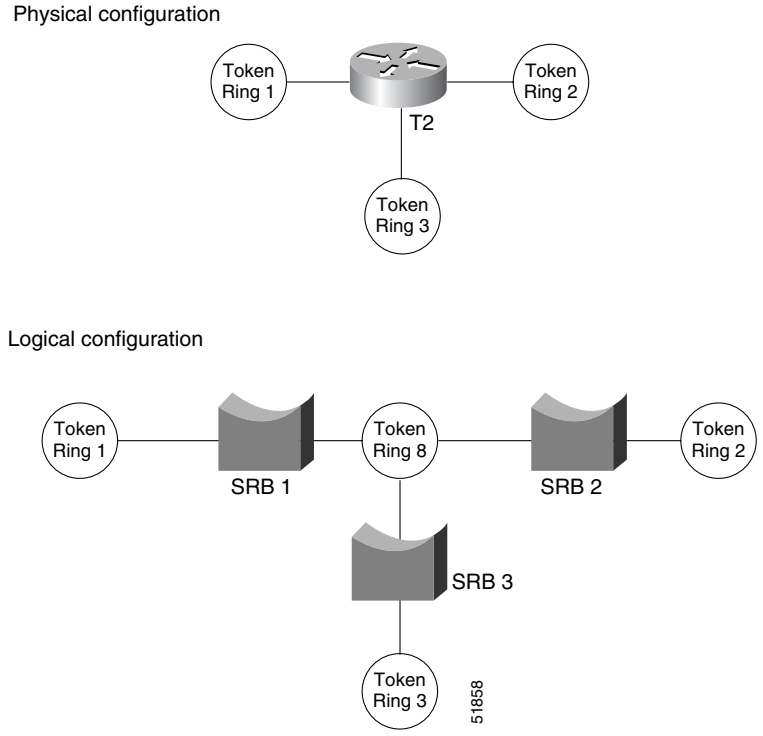
```
Wayfarer# show lnm config

Bridge(s) currently configured:
From     ring 001, address 0000.3000.abc4
Across bridge 002
To       ring 003, address 0000.3000.5735
```

In this example, the MAC addresses 0000.3000.abc4 and 000.3000.5735 must be configured as adapter addresses at the LNM Station.

LNM for a More Complex Network Example

Figure 44 shows a router with three Token Rings configured as a multiport bridge, thus employing the concept of the virtual ring.

Figure 44 Router with Three Token Rings Configured as a Multiport Bridge



The associated configuration file follows.

```
source-bridge ring-group 8
!
interface tokenring 0
 source-bridge 1 1 8
!
interface tokenring 1
 source-bridge 2 2 8
!
interface tokenring 2
 source-bridge 3 3 8
```

The **show lnm config** command displays the logical configuration of this bridge, including all the pertinent information for configuring this router into LNM:

```
Wayfarer# show lnm config

Bridge(s) currently configured:

            From     ring 001, address 0000.0028.abcd
            Across bridge 001
            To       ring 008, address 4000.0028.abcd

            From     ring 002, address 0000.3000.abc4
            Across bridge 002
            To       ring 008, address 4000.3000.abc4
```

```
From     ring 003, address 0000.3000.5735
Across bridge 003
To       ring 008, address 4000.3000.5735
```

In this example, six station definitions must be entered at the LNM Station, one for each of the MAC addresses listed in this sample **show lnm config** display.

NetBIOS Access Filters Example

The following command permits packets that include the station name ABCD to pass through the router, but denies passage to packets that do not include the station name ABCD:

```
netbios access-list host marketing permit ABCD
```

The following command specifies a prefix where the pattern matches any name beginning with the characters DEFG. Note that the string DEFG itself is included in this condition.

```
netbios access-list host marketing deny DEFG*
```

The following command permits any station name with the letter W as the first character and the letter Y as the third character in the name. The second and fourth letters in the name can be any character. This example would allow stations named WXYZ and WAYB; however, stations named WY and WXY would not be included in this statement, because the question mark must match some specific character in the name.

```
netbios access-list host marketing permit W?Y?
```

The following command illustrates how to combine wildcard characters:

```
netbios access-list host marketing deny AC?*
```

The command specifies that the marketing list deny any name beginning with AC that is at least three characters in length (the question mark would match any third character). The string ACBD and ACB would match, but the string AC would not.

The following command removes the entire marketing NetBIOS access list.

```
no netbios access-list host marketing
```

To remove single entries from the list, use a command such as the following:

```
no netbios access-list host marketing deny AC?*
```

This example removes only the list that filters station names with the letters AC at the beginning of the name.

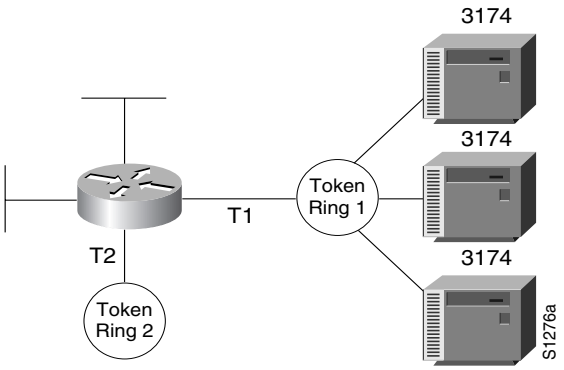
Access lists are scanned in order. In the following example, the first list denies all entries beginning with the letters ABC, including one named ABCD. This voids the second command, because the entry permitting a name with ABCD comes after the entry denying it.

```
netbios access-list host marketing deny ABC*
netbios access-list host marketing permit ABCD
```

Filtering Bridged Token Ring Packets to IBM Machines Example

The example in Figure 45 disallows the bridging of Token Ring packets to all IBM workstations on Token Ring 1.

Figure 45 Router Filtering Bridged Token Ring Packets to IBM Machines



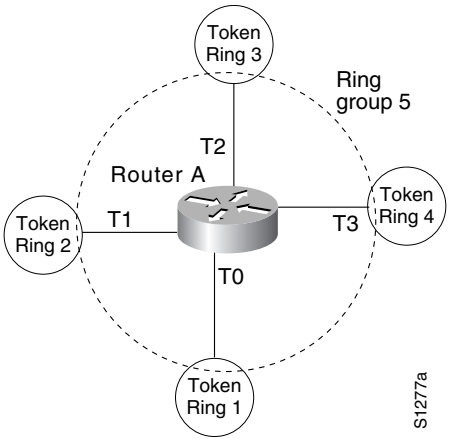
This example assumes that all hosts on Token Ring 1 have Token Ring addresses with the vendor code 1000.5A00.0000. The first line of the access list denies access to all IBM workstations, while the second line permits everything else. The access list is assigned to the input side of Token Ring 1.

```
! deny access to all IBM workstations
access-list 700 deny 1000.5A00.0000   8000.00FF.FFFF
! permit all other traffic
access-list 700 permit 0000.0000.0000   FFFF.FFFF.FFFF
!
interface token ring 1
! apply access list 700 to the input side of Token Ring 1
source-bridge input-address-list 700
```

Administrative Access Filters—Filtering SNAP Frames on Output Example

Figure 46 shows a router connecting four Token Rings.

Figure 46 Router Filtering SNAP Frames on Output



The following example allows only AppleTalk Phase 2 packets to be source-route bridged between Token Rings 0 and 1, and allows Novell packets only to be source-route bridged between Token Rings 2 and 3.

```
source-bridge ring-group 5
!
interface tokenring 0
 ip address 131.108.1.1 255.255.255.0
 source-bridge 1000 1 5
 source-bridge spanning
 source-bridge input-type-list 202
!
interface tokenring 1
 ip address 131.108.11.1 255.255.255.0
 source-bridge 1001 1 5
 source-bridge spanning
 source-bridge input-type-list 202
!
interface tokenring 2
 ip address 131.108.101.1 255.255.255.0
 source-bridge 1002 1 5
 source-bridge spanning
 source-bridge input-lsap-list 203
!
interface tokenring 3
 ip address 131.108.111.1 255.255.255.0
 source-bridge 1003 1 5
 source-bridge spanning
 source-bridge input-lsap-list 203
!
! SNAP type code filtering
! permit ATp2 data (0x809B)
! permit ATp2 AARP (0x80F3)
access-list 202 permit 0x809B 0x0000
access-list 202 permit 0x80F3 0x0000
access-list 202 deny 0x0000 0xFFFF


!
```

```
! LSAP filtering
! permit IPX (0xE0E0)
access-list 203 permit 0xE0E0 0x0101
access-list 203 deny 0x0000 0xFFFF
```

Note It is not necessary to check for an LSAP of 0xAAAA when filtering SNAP-encapsulated AppleTalk packets, because for source-route bridging, the use of type filters implies SNAP encapsulation.

Creating Access Filters Example

In math, you have the following:

```
3 x 4 + 2 = 14 but 3 x (4 + 2) = 18
```

Similarly, the following access expressions would return TRUE if lsap(201) and dmac(701) returned TRUE or if smac(702) returned TRUE:

```
lsap(201) & dmac(701) | smac(702)
```

However, the following access expression would return TRUE only if lsap(201) returned TRUE and either of dmac(701) or smac(702) returned TRUE:

```
lsap(201) & (dmac(701) | smac(702))
```

Referring to the earlier example, "An Example Using NetBIOS Access Filters," we had the phrase:

"Pass the frame if it is NetBIOS, or if it is an SNA frame destined to address 0110.2222.3333."

This phrase was converted to the simpler form of:

Pass if "NetBIOS or (SNA and destined to 0110.2222.3333)."

So, for the following configuration:

```
! Access list 201 passes NetBIOS frames (command or response)
access-list 201 permit 0xF0F0 0x0001
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
```

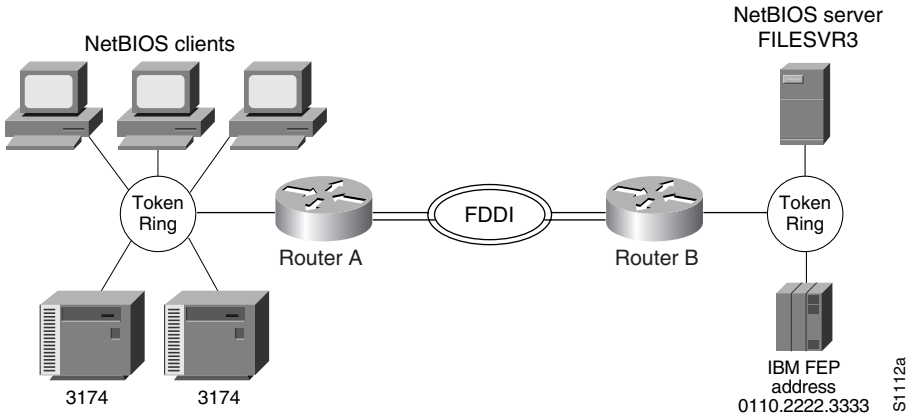
The following access expression would result:

```
access-expression in lsap(201) | (lsap(202) & dmac(701))
```

Access Filters Example

Figure 47 shows two routers connecting two Token Rings to an FDDI backbone.

Figure 47 Network Configuration Using NetBIOS Access Filters



Suppose you want to permit the IBM 3174 cluster controllers to access the FEP at address 0110.2222.3333, and also want the NetBIOS clients to access the NetBIOS server named FILESVR3. The following set of router configuration commands would meet this need:

```
netbios access-list host MIS permit FILESVR3
netbios access-list host MIS deny *
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
access-list 701 permit 0110.2222.3333
!
interface tokenring 0
access-expression in (lsap(202) & dmac(701)) | netbios-host(MIS)
```

Fast-Switching Example

The following example disables fast switching between two Token Ring interfaces in the same router. Frames entering Token Ring interfaces 0 or 1 will not be fast switched to the other interface.

```
! global command establishing the ring group for the interface configuration commands
source-bridge ring-group 2
!
! commands that follow apply to interface token 0
interface tokenring 0
! enable srb between local ring 1, bridge 1, and target ring 2
source-bridge 1 1 2
!disable source-route fast-switching cache on interface token 0
no source-bridge route-cache
!
interface token 1
! enable srb between local ring 2, bridge 1, and target ring 1
source-bridge 2 1 1
no source-bridge route-cache
```

Autonomous Switching Example

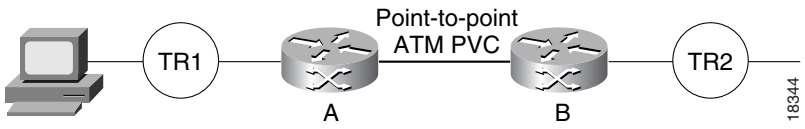
The following example enables use of autonomous switching between two ciscoBus Token Ring interfaces in the same router. Frames entering Token Ring interfaces 0 or 1 will be autonomously switched to the other interface.

```
! global command to apply interface configuration commands to the ring group
source-bridge ring-group 2
!
! commands that follow apply to interface token 0
interface tokenring 0
! enable srb between local ring 1, bridge 1, and target ring 2
source-bridge 1 1 2
! enable autonomous switching for interface token 0
source-bridge route-cache cbus
!
interface tokenring 1
! enable srb between local ring 2, bridge 1, and target ring 1
source-bridge 2 1 1
source-bridge route-cache cbus
```

Back-to-Back Routers ATM Configuration Example

Figure 48 shows a back-to-back scenario with two ATM adapters that are connected. There is no ATM switch in this example.

Figure 48 Connecting Routers Back-to-Back



Following are the configurations for routers A and B:

Router A

```
interface atm slot/port
 atm clock
interface atm slot/port.1 point-to-point
 atm pvc 1 10 12 aal5snap
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning
```

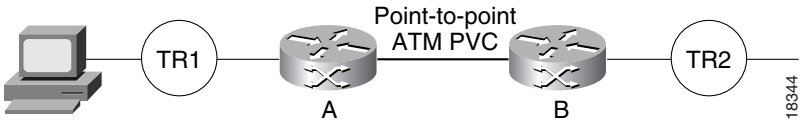
Router B

```
interface atm slot/port.1 point-to-point
 atm pvc 2 10 12 aal5snap
 source-bridge 100 1 200 conserve-ring
 source-bridge spanning
```

Single ATM PVC and Single Virtual Ring Per Router Configuration Example

Figure 49 shows an example with frames from Token Ring 1 destined to Token Ring 2 and an ATM switch connecting the routers.

Figure 49 Single ATM PVC and Single Virtual Ring Per Router



Router A

```
interface atm slot/port
interface atm slot/port.1 point-to-point
 atm pvc 1 10 12 aal5snap
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning
```

Router B

```
interface atm slot/port.1 point-to-point
 atm pvc 2 0 12 aal5snap
 source-bridge 100 1 200 conserve-ring
 source-bridge spanning
```

The following configuration does not use the **conserve-ring** argument in the configuration and the PVC is allocated its own virtual ring number.

Router A

```
source-bridge ring-group 100

interface atm slot/port
interface atm slot/port.1 point-to-point
 atm pvc 1 0 12 aal5snap
 source-bridge 5 1 100
 source-bridge spanning
```

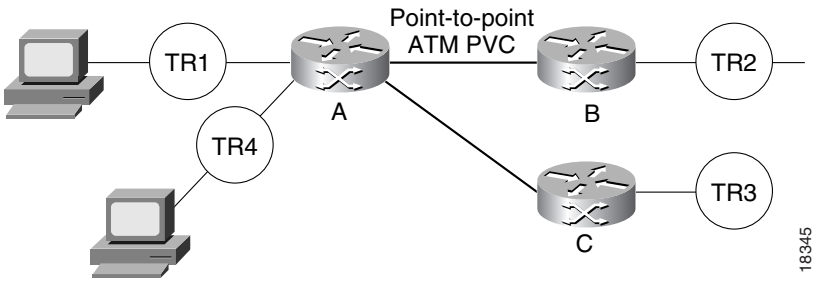
Router B

```
source-bridge ring-group 200

interface atm slot/port
interface atm slot/port.1 point-to-point
 atm pvc 2 0 12 aal5snap
 source-bridge 5 1 200
 source-bridge spanning
```

Multiple ATM PVCs and Multiple Virtual Rings on One Router Configuration Example

Figure 50 shows multiple ATM PVCs and multiple virtual rings on a router.

Figure 50 Multiple ATM PVCs and Multiple Virtual Rings on a Router



Following are the configurations for routers A, B, and C:

Router A

```
interface atm slot/port.1 point-to-point
 atm pvc 1 10 12 aal5snap
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning

interface atm slot/port.2 point-to-point
 atm 2 0 12 aal5snap
 source-bridge 300 2 101 conserve-ring
 source-bridge spanning
```

Router B

```
interface atm slot/port.1 point-to-point
 atm pvc 3 0 12 aal5snap
 source-bridge 100 1 200 conserve-ring
 source-bridge spanning
```

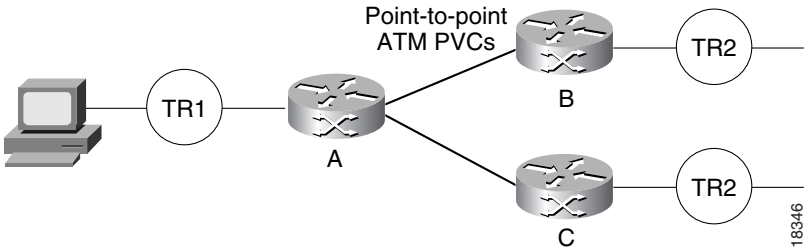
Router C

```
interface atm slot/port.1 point-to-point
 atm pvc 4 0 12 aal5snap
 source-bridge 101 2 300 conserve-ring
 source-bridge spanning
```

Multiple ATM PVCs with a Single Virtual Ring on the Router Configuration Example

Figure 51 shows traffic going from Token Ring 1 to Token Ring 2 and Token Ring 3.

Figure 51 Multiple ATM PVCs with a Single Virtual Ring on the Router



Following are the configurations for routers A, B, and C:

Router A

```
interface atm slot/port.1 point-to-point
 atm pvc 1 0 12 aal5snap
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning

interface atm slot/port.2 point-to-point
 atm pvc 2 0 2 aal5snap
 source-bridge 300 2 100 conserve-ring
 source-bridge spanning
```

Router B

```
interface atm slot/port.1 point-to-point
 atm pvc 3 0 2 aal5snap
 source-bridge 100 1 200 conserve-ring
 source-bridge spanning
```

Router C

```
interface atm slot/port.1 point-to-point
 atm pvc 4 1 3 aal5snap
 source-bridge 100 2 300 conserve-ring
 source-bridge spanning
```

Configuring Data-Link Switching Plus

This chapter describes how to configure data-link switching plus (DLSw+), Cisco's implementation of the DLSw standard for Systems Network Architecture (SNA) and NetBIOS devices. Refer to the *DLSw+ Design and Implementation Guide* for more complex configuration instructions. For a complete description of the DLSw+ commands mentioned in this chapter, refer to the "DLSw+ Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page lv in the "Using Cisco IOS Software" chapter.

Technology Overview

DLSw+ is a method of transporting SNA and NetBIOS. It complies with the DLSw standard documented in RFC 1795 and the DLSw Version 2 standard. DLSw+ is an alternative to RSRB that addresses several inherent problems that exist in RSRB, such as:

- SRB hop-count limits (SRB's limit is seven)
- Broadcast traffic (including SRB explorer frames or NetBIOS name queries)
- Unnecessary traffic (acknowledgments and keepalives)
- Data-link control timeouts

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

DLSw Standard

The DLSw standard, documented in RFC 1795, defines the switch-to-switch protocol between DLSw routers. The standard also defines a mechanism to terminate data-link control connections locally and multiplex the traffic from the data-link control connections to a TCP connection. The standard always calls for the transport protocol to be TCP and always requires that data-link control connections be locally terminated (the equivalent of Cisco's local acknowledgment option). The standard also requires that the SRB RIF be terminated at the DLSw router. The standard describes a means for prioritization and flow control and defines error recovery procedures that ensure data-link control connections are appropriately disabled if any part of their associated circuits breaks.

The DLSw standard does not specify when to establish TCP connections. The capabilities exchange allows compliance to the standard, but at different levels of support. The standard does not specify how to cache learned information about MAC addresses, RIFs, or NetBIOS names. It also does not describe how to track either capable or preferred DLSw partners for either backup or load-balancing purposes. The standard does not provide the specifics of media conversion, but leaves the details up to the implementation. It does not define how to map switch congestion to the flow control for data-link control. Finally, the MIB is documented under a separate RFC.

DLSw Version 2 Standard

In the Version 1 standard, a network design requires fully meshed connectivity so that all peers were connect to every other peer. This design creates unnecessary broadcast traffic because an explorer propagates to every peer for every broadcast.

The Version 2 standard is documented in RFC 2166. It includes RFC 1795 and adds the following enhancements:

- IP Multicast, page 2
- UDP Unicast, page 3
- Enhanced Peer-on-Demand Routing Feature, page 3
- Expedited TCP Connection, page 3

Users implement DLSw Version 2 for scalability if they are using multivendor DLSw devices with an IP multicast network. DLSw Version 2 requires complex planning because it involves configuration changes across an IP network.

IP Multicast

Multicast service avoids duplication and excessive bandwidth of broadcast traffic because it replicates and propagates messages to its multicast members only as necessary. It reduces the amount of network overhead in the following ways:

- Avoids the need to maintain TCP Switch-to-Switch Protocol (SSP) connections between two DLSw peers when no circuits are available
- Ensures that each broadcast results in only a single explorer over every link

DLSw Version 2 is for customers who run a multicast IP network and do not need the advantages of border peering.

UDP Unicast

DLSw Version 2 uses UDP unicast in response to an IP multicast. When address resolution packets (CANUREACH_EX, NETBIOS_NQ_ex, NETBIOS_ANQ, and DATAFRAME) are sent to multiple destinations (IP multicast service), DLSw Version 2 sends the response frames (ICANREACH_ex and NAME_RECOGNIZED_ex) via UDP unicast.

UDP unicast uses UDP source port 0. However, some firewall products treat packets that use UDP source port 0 as security violations, discarding the packets and preventing DLSw connections. To avoid this situation, use one of the following procedures:

- Configure the firewall to allow UDP packets to use UDP source port 0.
- Use the **dlsw udp-disable** command to disable UDP unicast and send address resolution packets in the existing TCP session.

Enhanced Peer-on-Demand Routing Feature

DLSw Version 2 establishes TCP connections only when necessary and the TCP connections are brought down when there are no circuits to a DLSw peer for a specified amount of time. This method, known as peer-on-demand routing, was recently introduced in DLSw Version 2, but has been implemented in Cisco DLSw+ border peer technology since Cisco IOS Release 10.3.

Expedited TCP Connection

DLSw Version 2 efficiently establishes TCP connections. Previously, DLSw created two unidirectional TCP connections and then disconnected one after the capabilities exchange took place. With DLSw Version 2, a single bidirectional TCP connection establishes if the peer is brought up as a result of an IP multicast/UDP unicast information exchange.

DLSw+ Features

DLSw+ is Cisco's version of DLSw and it supports several additional features and enhancements. DLSw+ is a means of transporting SNA and NetBIOS traffic over a campus or WAN. The end systems can attach to the network over Token Ring, Ethernet, Synchronous Data Link Control (SDLC) Protocol, Qualified Logical Link Control (QLLC), or Fiber Distributed Data Interface (FDDI). See the *DLSw+ Design and Implementation Guide* Appendix B, "DLSw+ Support Matrix," for details. DLSw+ switches between diverse media and locally terminates the data links, keeping acknowledgments, keepalives, and polling off the WAN. Local termination of data links also eliminates data-link control timeouts that can occur during transient network congestion or when rerouting around failed links. Finally, DLSw+ provides a mechanism for dynamically searching a network for SNA or NetBIOS resources and includes caching algorithms that minimize broadcast traffic.

DLSw+ is fully compatible with any vendor's RFC 1795 implementation and the following features are available when both peers are using DLSw+:

- Peer groups and border peers
- Backup peers
- Promiscuous and on-demand peers
- Explorer firewalls and location learning
- NetBIOS dial-on-demand routing feature support

- UDP unicast support
- Load balancing
- Support for LLC1 circuits
- Support for multiple bridge groups
- Support for RIF Passthrough
- SNA type of service feature support
- Local acknowledgment for Ethernet-attached devices and media conversion for SNA PU 2.1 and PU 2.0 devices
- Conversion between LLC2 to SDLC between PU 4 devices
- Local or remote media conversion between LANs and either SDLC Protocol or QLLC
- SNA View, Blue Maps, and Internetwork Status Monitor (ISM) support

MIB enhancements that allow DLSw+ features to be managed by the CiscoWorks Blue products, SNA Maps, and SNA View. Also, new traps alert network management stations of peer or circuit failures. For more information, refer to the current Cisco IOS release note for the location of the Cisco MIB website.

Local Acknowledgment

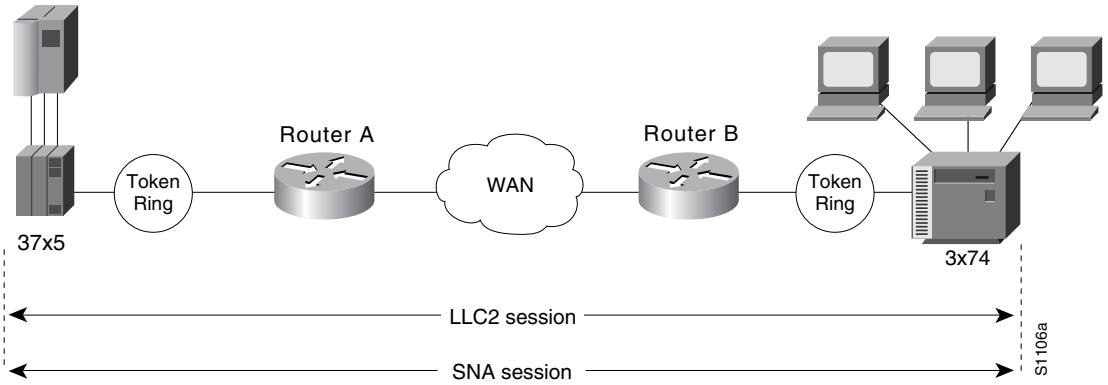
When you have LANs separated by wide geographic distances, and you want to avoid sending data multiple times, and the loss of user sessions that can occur with time delays, encapsulate the source-route bridged traffic inside IP datagrams passed over a TCP connection between two routers with local acknowledgment enabled.

Logical Link Control, type 2 (LLC2) is an ISO standard data-link level protocol used in Token Ring networks. LLC2 was designed to provide reliable sending of data across LAN media and to cause minimal or at least predictable time delays. However, DLSw+ and WAN backbones created LANs that are separated by wide, geographic distances-spanning countries and continents. As a result, LANs have time delays that are longer than LLC2 allows for bidirectional communication between hosts. Local acknowledgment addresses the problem of unpredictable time delays, multiple sendings, and loss of user sessions.

In a typical LLC2 session, when one host sends a frame to another host, the sending host expects the receiving host to respond positively or negatively in a predefined period of time commonly called the *T1 time*. If the sending host does not receive an acknowledgment of the frame it sent within the T1 time, it retries a few times (normally 8 to 10). If there is still no response, the sending host drops the session.

Figure 127 illustrates an LLC2 session in which a 37x5 on a LAN segment communicates with a 3x74 on a different LAN segment separated via a wide-area backbone network. Frames are transported between Router A and Router B by means of DLSw+. However, the LLC2 session between the 37x5 and the 3x74 is still end-to-end; that is, every frame generated by the 37x5 traverses the backbone network to the 3x74, and the 3x74, on receipt of the frame, acknowledges it.

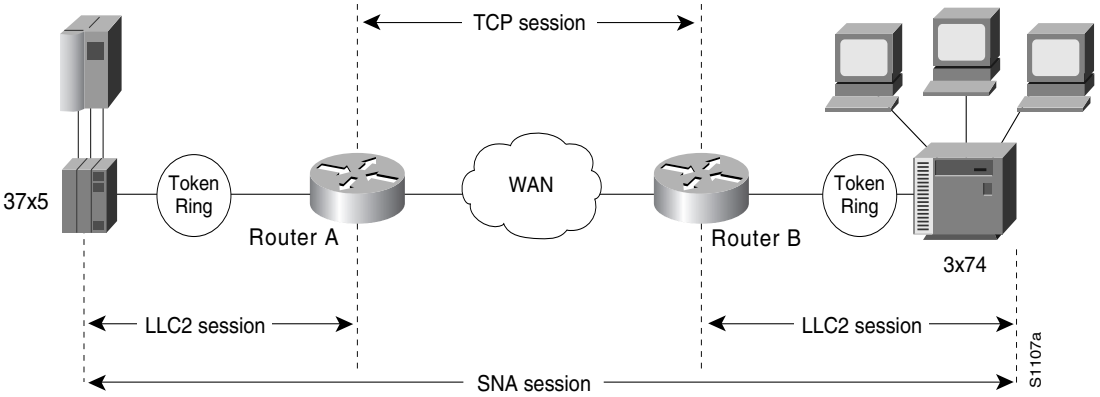
Figure 127 LLC2 Session without Local Acknowledgment



On backbone networks consisting of slow serial links, the T1 timer on end hosts could expire before the frames reach the remote hosts, causing the end host to resend. Resending results in duplicate frames reaching the remote host at the same time as the first frame reaches the remote host. Such frame duplication breaks the LLC2 protocol, resulting in the loss of sessions between the two IBM machines.

One way to solve this time delay is to increase the timeout value on the end nodes to account for the maximum transit time between the two end machines. However, in networks consisting of hundreds or even thousands of nodes, every machine would need to be reconfigured with new values. With local acknowledgment for LLC2 enabled, the LLC2 session between the two end nodes would not be not end-to-end, but instead, would terminate at two local routers. Figure 128 shows the LLC2 session with the 37x5 ending at Router A and the LLC2 session with the 3x74 ending at Router B. Both Router A and Router B execute the full LLC2 protocol as part of local acknowledgment for LLC2.

Figure 128 LLC2 Session with Local Acknowledgment



With local acknowledgment for LLC2 enabled in both routers, Router A acknowledges frames received from the 37x5. The 37x5 still operates as if the acknowledgments it receives are from the 3x74. Router A looks like the 3x74 to the 37x5. Similarly, Router B acknowledges frames received from the 3x74. The 3x74 operates as if the acknowledgments it receives are from the 37x5. Router B looks like the 3x74 to 37x5. Because the frames do not have to travel the WAN backbone networks to be acknowledged, but are locally acknowledged by routers, the end machines do not time out, resulting in no loss of sessions.

Enabling local acknowledgment for LLC2 has the following advantages:

- Local acknowledgment for LLC2 solves the T1 timer problem without having to change any configuration on the end nodes. The end nodes are unaware that the sessions are locally acknowledged. In networks consisting of hundreds or even thousands of machines, this is a definite advantage. All the frames acknowledged by the Cisco IOS software appear to the end hosts to be coming from the remote IBM machine. In fact, by looking at a trace from a protocol analyzer, one cannot say whether a frame was acknowledged by the local router or by a remote IBM machine. The MAC addresses and the RIFs generated by the Cisco IOS software are identical to those generated by the remote IBM machine. The only way to find out whether a session is locally acknowledged is to use either a **show local-ack** command or a **show source-bridge** command on the router.
- All the supervisory (RR, RNR, REJ) frames that are locally acknowledged go no farther than the router. Without local acknowledgment for LLC2, *every* frame traverses the backbone.

  With local acknowledgment, only data (I-frames) traverse the backbone, resulting in less traffic on the backbone network. For installations in which customers pay for the amount of traffic passing through the backbone, this could be a definite cost-saving measure. A simple protocol exists between the two *peers* to bring up or down a TCP session.

Notes on Using LLC2 Local Acknowledgment

LLC2 local acknowledgment is enabled with TCP and DLSw+ Lite remote peers.

If the LLC2 session between the local host and the router terminates in either router, the other will be informed to terminate its connection to its local host.

If the TCP queue length of the connection between the two routers reaches the high-water mark, the routers sends Receiver-Not-Ready (RNR) messages to the local hosts until the queue limit is reduced to below this limit. It is possible, however, to prevent the RNR messages from being sent by using the **dlsw llc2 nornr** command.

The configuration of the LLC2 parameters for the local Token Ring interfaces can affect overall performance. Refer to the chapter “Configuring LLC2 and SDLC Parameters” in this manual for more details about fine-tuning your network through the LLC2 parameters.

The routers at each end of the LLC2 session execute the full LLC2 protocol, which could result in significant router overhead. The decision to use local acknowledgment for LLC2 should be based on the speed of the backbone network in relation to the Token Ring speed. For LAN segments separated by slow-speed serial links (for example, 56 kbps), the T1 timer problem could occur more frequently. In such cases, it might be wise to turn on local acknowledgment for LLC2. For LAN segments separated by a T1, backbone delays will be minimal; in such cases, DLSw+, FST or direct encapsulation should be considered in order to disable local acknowledgement. Speed mismatch between the LAN segments and the backbone network is one criterion to help you decide to use local acknowledgment for LLC2.

There are some situations (such as the receiving host failing between the time the sending host sends data and the time the receiving host receives it), in which the sending host would determine, *at the LLC2 layer*, that data was received when it actually was not. This error occurs because the router acknowledges that it received data from the sending host before it determines that the receiving host can actually receive the data. But because both NetBIOS and SNA have error recovery in situations where an end device goes down, these higher-level protocols will resend any missing or lost data. Because these transaction request/confirmation protocols exist above LLC2, they are not affected by tight timers, as is LLC2. They also are transparent to local acknowledgment.

If you are using NetBIOS applications, note that there are two NetBIOS timers—one at the link level and one at the next higher level. Local acknowledgment for LLC2 is designed to solve link timeouts only. If you are experiencing NetBIOS session timeouts, you have two options:

- Experiment with increasing your NetBIOS timers and decreasing your maximum NetBIOS frame size.
- Avoid using NetBIOS applications on slow serial lines.

Note By default, the Cisco IOS software translates Token Ring LLC2 to Ethernet 802.3 LLC2. To configure the router to translate Token Ring LLC2 frames into Ethernet 0x80d5 format frames, refer to the section "Enable Token Ring LLC2-to-Ethernet Conversion" in the "Configuring Source-Route Bridging" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

DLSw+ Support for Other SNA Features

DLSw+ can be used as a transport for SNA features such as LNM, DSPU, SNA service point, and SNA Switching Services (SNASw) through a Cisco IOS feature called virtual data-link control (VDLC).

LNM over DLSw+ allows DLSw+ to be used in Token Ring networks that are managed by IBM's LNM software. Using this feature, LNM can be used to manage Token Ring LANs, control access units, and Token Ring attached devices over a DLSw+ network. All management functions continue to operate as they would in a source-route bridged network or an RSRB network.

DSPU over DLSw+ allows Cisco's DSPU feature to operate in conjunction with DLSw+ in the same router. DLSw+ can be used either upstream (toward the mainframe) or downstream (away from the mainframe) of DSPU. DSPU concentration consolidates the appearance of multiple PUs into a single PU appearance to VTAM, minimizing memory and cycles in central site resources (VTAM, NCP, and routers) and speeding network startup.

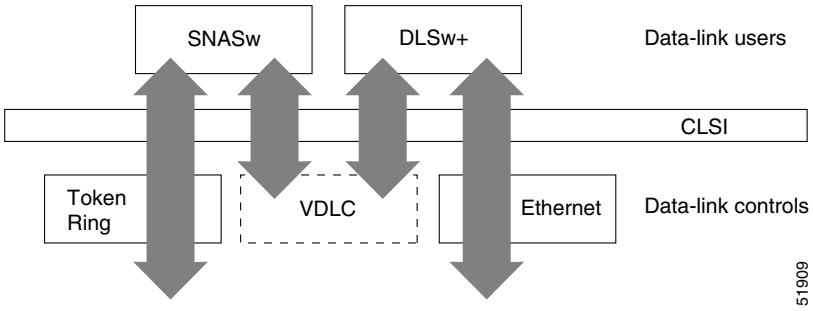
SNA service point over DLSw+ allows Cisco's SNA service point feature to be used in conjunction with DLSw+ in the same router. Using this feature, SNA service point can be configured in remote routers, and DLSw+ can provide the path for the remote service point PU to communicate with NetView. This allows full management visibility of resources from a NetView 390 console, while concurrently offering the value-added features of DLSw+ in an SNA network.

SNASw over DLSw+ allows Cisco's APPN Branch Extender functionality to be used in conjunction with DLSw+ in the same router. With this feature, DLSw+ can be used to access SNASw in the data center. DLSw+ can also be used as a transport for SNASw upstream connectivity, providing nondisruptive recovery from failures.

Using DLSw+ as a transport for other Cisco IOS SNA features requires a feature called VDLC. Cisco IOS data-link users (such as LNM, DSPU, SNA service point, and SNASw) write to a virtual data-link control interface. DLSw+ then reads from this interface and sends out the traffic. Similarly, DLSw+ can receive traffic destined for one of these data-link users and write it to the virtual data-link control interface, from which the appropriate data-link user will read it.

In Figure 129, SNASw and DLSw+ use Token Ring and Ethernet, respectively, as "real" data-link controls, and use virtual data-link control to communicate between themselves. When one of the high-layer protocols passes data to the virtual data-link control, the virtual data-link control must pass it to a higher-layer protocol; nothing leaves the virtual data-link control without going through a data-link user.

Figure 129 VDLC Interaction with Higher-Layer Protocols



The higher-layer protocols make no distinction between the VDLC and any other data-link control, but they do identify the VDLC as a destination. In the example shown in Figure 129, SNASw has two ports: a physical port for Token Ring and a virtual port for the VDLC. When you define the SNASw VDLC port, you can specify the MAC address assigned to it. Data transport from SNASw to DLSw+ by way of the VDLC is directed to the VDLC MAC address. The type of higher-layer protocol you use determines how the VDLC MAC address is assigned.

DLSw+ Configuration Task List

DLSw+ supports local or remote media conversion between LANs and SDLC or QLLC.

To configure DLSw+, complete the tasks in the following sections:

- Defining a DLSw+ Local Peer for the Router, page 8
- Defining a DLSw+ Remote Peer, page 9
- Mapping DLSw+ to a Local Data-Link Control, page 12
- Configuring Advanced Features, page 15
- Configuring DLSw+ Timers, page 30

See the "DLSw+ Configuration Examples" section on page 32 for examples.

Defining a DLSw+ Local Peer for the Router

Defining a DLSw+ local peer for a router enables DLSw+. Specify all local DLSw+ parameters as part of the local peer definition. To define a local peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw local peer [peer-id ip-address] [group group] [border] [cluster cluster-id] [cost cost] [lf size] [keepalive seconds] [passive] [promiscuous] [init-pacing-window size] [max-pacing-window size] [biu-segment]`	Defines the DLSw+ local peer.

The following is a sample **dlsw local peer** statement:

```
dlsw local peer peer-id 10.2.34.3
```

Defining a DLSw+ Remote Peer

Defining a remote peer in DLSw+ is optional, however, usually at least one side of a peer connection has a **dlsw remote-peer** statement. If you omit the **dlsw remote-peer** command from a DLSw+ peer configuration, then you must configure the **promiscuous** keyword on the dlsw local-peer statement. Promiscuous routers will accept any peer connection requests from other routers that are not preconfigured. To define a remote peer, use the **dlsw remote-peer** command in global configuration mode.

One of the options in the remote peer statement is to specify an encapsulation type. Configure one of the following types of encapsulations with the **dlsw remote-peer** statement:

- TCP Encapsulation, page 9
- TCP/IP with RIF Passthrough Encapsulation, page 10
- FST Encapsulation, page 10
- Direct Encapsulation, page 11
- DLSw Lite Encapsulation, page 11

Which encapsulation type you choose depends on several factors, including whether you want to terminate the LLC flows. TCP and DLSw+ Lite terminate the LLC, but the other encapsulation types do not. For details on each encapsulation type, see the *DLSw+ Design and Implementation Guide*. See the "Local Acknowledgement" section in the overview chapter of this publication for a discussion on local acknowledgement.

TCP Encapsulation

To configure TCP encapsulation on a remote peer, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **tcp** *ip-address* [[*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** bytes-list-name] [**circuit-weight** *weight*] [**cluster** *cluster-id*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**inactivity**] [**dynamic**] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**] [**rif-passthru** virtual-ring-number] [**tcp-queue-max** *size*] [**timeout** *seconds*]	Defines a remote peer with TCP encapsulation.

The following command specifies a **dlsw remote peer** with TCP encapsulation:

```
dlsw remote-peer 0 tcp 10.23.4.5
```

TCP/IP with RIF Passthrough Encapsulation

To configure TCP/IP with RIF Passthrough encapsulation, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **tcp** *ip-address* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**inactivity**] [**dynamic**] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**] [**rif-passthru** *virtual-ring-number*] [**tcp-queue-max** *size*] [**timeout** *seconds*]	Defines a remote peer with TCP/IP with RIF Passthrough encapsulation.

The following command specifies a remote peer with TCP/IP with RIF Passthrough encapsulation:

```
dlsw remote-peer 0 tcp 10.2.23.5 rif-passthru 100
```

FST Encapsulation

To configure FST encapsulation on a remote peer, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **fst** *ip-address* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*]	Defines a remote peer with FST encapsulation.

The following command specifies a DLSw remote peer with FST encapsulation:

```
dlsw remote-peer 0 fst 10.2.23.5
```

Direct Encapsulation

To configure direct encapsulation, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **frame-relay interface serial** *number dlci-number* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** host-list-name] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] **pass-thru**	Defines a remote peer with direct encapsulation.

Direct encapsulation is supported over High-Level Data Link Control (HDLC) and Frame Relay.

The following command specifies a DLSw remote peer with direct encapsulation over HDLC:

```
dlsw remote-peer 0 interface serial 01
```

Direct encapsulation over Frame Relay comes in two forms: DLSw Lite (LLC2 encapsulation) and Passthrough. Specifying the **pass-thru** option configures the router so that the traffic will not be locally acknowledged. (DLSw+ normally locally acknowledges traffic to keep traffic on the WAN to a minimum.)

The following command specifies a DLSw remote peer with Direct encapsulation with pass-thru over Frame Relay:

```
dlsw remote-peer 0 frame-relay interface serial 01 pass-thru
```

DLSw+ Lite is described in the "DLSw Lite Encapsulation" section on page 11.

DLSw Lite Encapsulation

To configure DLSw Lite encapsulation, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **frame-relay interface serial** *number dlci-number* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] **pass-thru**	Defines a remote peer with DLSw Lite encapsulation.

The following command specifies a DLSw remote peer with DLSw Lite encapsulation over Frame Relay:

```
dlsw remote-peer 0 frame-relay interface serial 01
```

Mapping DLSw+ to a Local Data-Link Control

In addition to configuring local and remote peers, you must map one of the following local data-link controls to DLSw+:

Token Ring

Traffic that originates from Token Ring is source-route bridged from the local ring onto a source-bridge ring group and then picked up by DLSw+. You must include a **source-bridge ring-group** command that specifies a virtual ring number when configuring Token Ring with DLSw+. In addition, you must configure the **source-bridge** command that tells the DLSw+ router to bridge from the physical Token Ring to the virtual ring.

To specify a virtual ring number, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge ring-group` *`ring-group [virtual-mac-address]`*	Defines a virtual ring.

To enable DLSw+ to bridge from the physical Token Ring ring to the virtual ring, use the following command in interface mode:

Command	Purpose
`Router(config-if)# source-bridge` *`source-ring-number bridge-number target-ring-number`*	Defines SRB on interface.

To enable single-route explorers, use the following command in interface mode:

Command	Purpose
`Router(config-if)# source-bridge` *`spanning`*	Enables single-route explorers.

Configuring the **source-bridge spanning** command is required because DLSw+ uses single-route explorers by default.

The following command configures a source-bridge ring-group and a virtual ring with a value of 100 to DLSw+:

```
source-bridge ring-group 100
int T0
 source-bridge 1 1 100
 source-bridge spanning
```

The *ring-group* number specified in the **source-bridge** command must be the number of a defined source-bridge ring-group or DLSw+ will not see this interface.

Ethernet

Traffic that originates from Ethernet is picked up from the local Ethernet interface bridge group and transported across the DLSw+ network. Therefore, you must map a specific Ethernet bridge group to DLSw+.

To map an Ethernet bridge group to DLSw+, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw bridge-group** *group-number* [**llc2** [**N2** *number*] [**ack-delay-time** *milliseconds*] [**ack-max** *number*] [**idle-time** *milliseconds*] [**local-window** *number*] [**t1-time** *milliseconds*] [**tbusy-time** *milliseconds*] [**tpf-time** *milliseconds*] [**trej-time** *milliseconds*] [**txq-max** *number*] [**xid-neg-val-time** *milliseconds*] [**xid-retry-time** *milliseconds*]] [**locaddr-priority** *lu address priority list number*] [**sap-priority** *priority list number*]	Links DLSw+ to the bridge group of the Ethernet LAN.

To assign the Ethernet interface to a bridge group, use the following command in interface configuration mode:

Command	Purpose
Router(config-if)# **bridge-group** *bridge-group*	Assigns the Ethernet interface to a bridge group.

The following command maps bridge-group 1 to DLSw+:

```
dlsw bridge-group 1
int E1
 bridge-group 1
bridge 1 protocol ieee
```

SDLC

Configuring SDLC devices is more complicated than configuring Ethernet and Token Ring. There are several considerations that affect which interface commands are configured. See the *DLSw+ Design and Implementation Guide* for more details.

To establish devices as SDLC stations, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	Router(config-if)# **encapsulation sdlc**	Sets the encapsulation type of the serial interface to SDLC.
Step 2	Router(config-if)# **sdlc role** {**none** | **primary** | **secondary** | **prim-xid-poll**}	Establishes the role of the interface.
Step 3	Router(config-if)# **sdlc vmac** *mac-address*[1]	Configures a MAC address for the serial interface.
Step 4	Router(config-if)# **sdlc address** *hexbyte* [**echo**]	Assigns a set of secondary stations attached to the serial link.
Step 5	Router(config-if)# **sdlc partner** *mac-address sdlc-address* {**inbound** | **outbound**}	Specifies the destination address with which an LLC session is established for the SDLC station.

	Command	Purpose
Step 6	`Router(config-if)# sdlc xid`	Specifies an XID value appropriate for the designated SDLC station associated with this serial interface.
Step 7	`Router(config-if)# sdlc dlsw {sdlc-address | default | partner mac-address [inbound | outbound]}`	Enables DLSw+ on an SDLC interface.

1. The last byte of the MAC address must be 00.

Use the **default** option if you have more than 10 SDLC devices to attach to the DLSw+ network. To configure an SDLC multidrop line downstream, you configure the SDLC role as either **primary** or **prim-xid-poll**. SDLC role **primary** specifies that any PU without the **xid-poll** parameter in the **sdlc address** command is a PU 2.0 device. SDLC role **prim-xid-poll** specifies that every PU is type 2.1. We recommend that you specify **sdlc role primary** if all SDLC devices are type PU 2.0 or a mix of PU 2.0 and PU 2.1. Specify **sdlc role prim-xid-poll** if all devices are type PU 2.1.

To configure DLSw+ to support LLC2-to-SDLC conversion for PU 4 or PU 5 devices, specify the **echo** option in the **sdlc address** command. A PU 4-to-PU 4 configuration requires that **none** be specified in the **sdlc role** command.

Refer to the "DLSw+ with SDLC Multidrop Support Configuration Examples" section on page 38 and the "DLSw+ with LLC2-to-SDLC Conversion Between PU 4-to-PU 4 Communication Example" section on page 39 for sample configurations.

The following configuration shows a DLSw+ router configured for SDLC:

```
dlsw local-peer peer-id 10.2.2.2
dlsw remote-peer 0 tcp 10.1.1.1
interface Serial1
mtu 6000
no ip address
encapsulation sdlc
no keepalive
nrzi-encoding
clockrate 9600
sdlc vmac 4000.3745.0000
sdlc N1 48016
sdlc address 04 echo
sdlc partner 4000.1111.0020 04
sdlc dlsw 4
```

QLLC

SNA devices use QLLC when connecting to X.25 networks. QLLC essentially emulates SDLC over x.25. Therefore, configuring QLLC devices is also complicated. There are several considerations that affect which interface commands are configured. See the *DLSw+ Design and Implementation Guide* for details.

You can configure DLSw+ for QLLC connectivity, which enables both of the following scenarios:

- Remote LAN-attached devices (physical units) or SDLC-attached devices can access an FEP or an AS/400 over an X.25 network.

  Our QLLC support allows remote X.25-attached SNA devices to access an FEP without requiring X.25 NCP Packet Switching Interface (NPSI) in the FEP. This may eliminate the requirement for NPSI (if GATE and DATE are not required), thereby eliminating the recurring license cost. In addition, because the QLLC attached devices appear to be Token Ring-attached to the Network Control Program (NCP), they require no preconfiguration in the FEP. Remote X.25-attached SNA devices can also connect to an AS/400 over Token Ring using this support.

- Remote X.25-attached SNA devices can access an FEP or an AS/400 over a Token Ring or over SDLC.

  For environments just beginning to migrate to LANs, our QLLC support allows deployment of LANs in remote sites while maintaining access to the FEP over existing NPSI links. Remote LAN-attached devices (physical units) or SDLC-attached devices can access a FEP over an X.25 network without requiring X.25 hardware or software in the LAN-attached devices. The Cisco IOS software supports direct attachment to the FEP over X.25 without the need for routers at the data center for SNA traffic.

To enable QLLC connectivity for DLSw+, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# encapsulation x 25`	Specifies an interface as an X.25 device.
Step 2	`Router(config-if)# x25 address` *subaddress*	Activates X.25 subaddresses.
Step 3	`Router(config-if)# x25 map qllc` *virtual-mac-addr x121-addr* `[cud` *cud-value*`] [`*x25-map-options*`]`	Associates a virtual MAC address with the X.121 address of the remote X.25 device.
Step 4	`Router(config-if)# qllc dlsw {subaddress` *subaddress* `| pvc` *pvc-low* `[`*pvc-high*`]} [vmac` *vmacaddr* `[`*poolsize*`]] [partner` partner-*macaddr*`] [sap` *ssap dsap*`] [xid` *xidstring*`] [npsi-poll]`	Enables DLSw+ over QLLC.

The following configuration enables QLLC connectivity for DLSw+:

```
dlsw local-peer peer-id 10.3.12.7
dlsw remote-peer 0 tcp 10.3.1.4
interface S0
 encapsulation x25
 x25 address 3110212011
 x25 map qllc 1000.0000.0001 3 1104150101
 qllc dlsw partner 4000.1151.1234
```

FDDI

Configure an FDDI interface the same as a Token Ring or Ethernet interface, depending on whether you are configuring SRB or Transparent Bridging. If you are configuring the router for SRB, configure the FDDI interface for Token Ring. If you are configuring the router for Transparent Bridging, configure the FDDI interface for Ethernet.

Configuring Advanced Features

DLSw+ goes beyond the standard to include additional functionality in the following areas:

- Scalability, page 16—Constructs IBM internetworks in a way that reduces the amount of broadcast traffic, which enhances their scalability.
- Availability, page 23—Dynamically finds alternate paths and, optionally, load-balances across multiple active peers, ports, and channel gateways.

- Modes of Operation, page 26—Dynamically detects the capabilities of the peer router and operates according to those capabilities.
- Network Management, page 27—Works with enhanced network management tools such as CiscoWorks Blue Maps, CiscoWorks SNA View, and CiscoWorks Blue Internetwork Status Monitor (ISM).
- Traffic Bandwidth and Queueing Management, page 27—Offers several bandwidth management and queueing features to enhance the overall performance of your DLSw+ network. Controls different types of explorer traffic using multiple queues, each with a wide range of depth settings.
- Access Control, page 27—Provides access control to various resources throughout a network.

Scalability

One significant factor that limits the size of Token Ring internet works is the amount of explorer traffic that traverses the WAN. DLSw+ includes the following features to reduce the number of explorers:

- Peer Groups and Border Peers, page 16
- Explorer Firewalls, page 20
- NetBIOS Dial-on-Demand Routing, page 20
- SNA Dial-on-Demand Routing, page 21
- UDP Unicast Feature, page 21
- LLC1 Circuits, page 22
- Dynamic Peers, page 22
- Promiscuous Peer Defaults, page 22

Peer Groups and Border Peers

Perhaps the most significant optimization in DLSw+ is a feature known as *peer groups*. Peer groups are designed to address the broadcast replication that occurs in a fully meshed network. When any-to-any communication is required (for example, for NetBIOS or Advanced Peer-to-Peer Networking [APPN] environments), RSRB or standard DLSw implementations require peer connections between every pair of routers. This setup is not only difficult to configure, but it results in branch access routers having to replicate search requests for each peer connection. This setup wastes bandwidth and router cycles. A better concept is to group routers into clusters and designate a focal router to be responsible for broadcast replication. This capability is included in DLSw+.

With DLSw+, a cluster of routers in a region or a division of a company can be combined into a peer group. Within a peer group, one or more of the routers is designated to be the *border peer*. Instead of all routers peering to one another, each router within a group peers to the border peer; and border peers establish peer connections with each other. When a DLSw+ router receives a TEST frame or NetBIOS NAME-QUERY, it sends a single explorer frame to its border peer. The DLSw+ border peer router checks its local, remote and group cache for any reachability information before forwarding the explorer. If no match is found, the border peer forwards the explorer on behalf of the peer group member. If a match is found, the border peer sends the explorer to the appropriate peer or border peer. This setup eliminates duplicate explorers on the access links and minimizes the processing required in access routers.

You can further segment DLSw+ routers within the same border peer group that are serving the same LANs into a *peer cluster*. This segmentation reduces explorers because the border peer recognizes that it only has to forward an explorer to one member within a *peer cluster*. Only TCP encapsulation can be used with the DLSw+ Peer Clusters feature.

The DLSw+ Peer Clusters feature is configured locally on the member peer or on a border peer. Although both options can be configured, we recommend that the *cluster-id* of a particular peer is defined in either the border peer or on the member peer, but not both because of potential configuration confusion.

To define peer groups, configure border peers and assign the local peer to a peer cluster, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw local-peer** [**peer-id** *ip-address*] [**group** *group*] [**border**] [**cost** *cost*] [**cluster** *cluster-id*] [**lf** *size*] [**keepalive** *seconds*] [**passive**] [**promiscuous**] [**biu-segment**] [**init-pacing-window** *size*] [**max-pacing-window** *size*]	Enables peer groups and border peers.

Use the **group** keyword to define a peer group, the **border** keyword to define a border peer and the **cluster** keyword to assign the local peer to a peer cluster. When the user defines the **cluster** option in the **dlsw local-peer** command on the member peer router, the cluster information is exchanged with the border peer during the capabilities exchange as the peers become active. The border peer uses this information to make explorer replication and forwarding decisions.

The following command configures the router as the Border peer that is a member of group 2:

```
dlsw local-peer peer-id 10.2.13.4 group 2 border
```

Configure the **cluster** option in the **dlsw remote-peer** command on a border peer to enable the DLSw+ Peer Clusters feature without forcing every DLSw+ router in the network to upgrade their software. To enable the DLSw+ Peer Clusters feature on a Border Peer, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **tcp** *ip-address* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cluster** *cluster-id*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**inactivity**] [**dynamic**] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**] [**rif-passthru** *virtual-ring-number*] [**tcp-queue-max** *size*] [**timeout** *seconds*]	Defines the border peer router as part of a particular cluster and enables the DLSw+ Peer Clusters feature.

The following command configures a border router as a member of cluster 5:

```
dlsw remote-peer tcp 10.2.13.5 cluster 5
```

A peer-on-demand peer is a non-configured remote-peer that was connected because of an LLC2 session established through a border peer DLSw+ network. On-demand peers greatly reduce the number of peers that must be configured. You can use on-demand peers to establish an end-to-end circuit even though the DLSw+ routers servicing the end systems have no specific configuration information about the peers. This configuration permits casual, any-to-any connection without the burden of configuring the connection in advance. It also allows any-to-any switching in large internetworks where persistent TCP connections would not be possible.

To configure peer-on-demand defaults, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw peer-on-demand-defaults [fst] [bytes-netbios-out bytes-list-name] [cost cost] [dest-mac destination mac-address] [dmac-output-list access-list-number] [host-netbios-out host-list-name] [inactivity minutes] [keepalive seconds] [lf size] [lsap-output-list list] [port-list port-list-number] [priority] [tcp-queue-max]`	Configures peer-on-demand defaults.

To define the maximum entries maintained in a border peer's group cache, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw group-cache max-entries number`	Defines the maximum entries in a group cache.

To remove all entries from the DLSw+ reachability cache, use the following command in privileged EXEC mode:

Command	Purpose
`Router# clear dlsw reachability`	Removes all entries from the DLSw+ reachability cache.

To reset to zero the number of frames that have been processed in the local, remote and group caches, use the following command in privileged EXEC mode:

Command	Purpose
`Router# clear dlsw statistics`	Resets to zero the number of frames that have been processed in the local, remote, and group caches.

To disable the border peer caching feature, use the following command in global configuration mode:

Command	Purpose
`Router(config-if)# dlsw group-cache disable`	Disables the border peer caching feature.

To verify that the peer cluster feature is enabled or that the border peer is configured, issue the **show dlsw capabilities** command on the router. To verify the cluster id number of which the peer is a member, issue the **show dlsw capabilities local** command on the local router.

To display the contents of the reachability caches, use the following command in privileged EXEC command mode:

Command	Purpose
`Router# show dlsw reachability [[group [value] | local | remote] | [mac-address [address] [netbios-names [name]`	Displays content of group, local and remote caches.

Use the **group** keyword to display the reachability information for the border peer.

Explorer Firewalls

An explorer firewall permits only a single explorer for a particular destination MAC address or NetBIOS name to be sent across the WAN. While an explorer is outstanding and awaiting a response from the destination, subsequent explorers for that MAC address or NetBIOS name are merely stored. When the explorer response is received at the originating DLSw+, all explorers receive an immediate local response. This eliminates the start-of-day explorer storm that many networks experience. Configure the **dlsw timer** command to enable explorer firewalls. See the "Configuring DLSw+ Timers" section on page 30 for details of the command.

To enable explorer firewalls, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw timer {icannotreach-block-time | netbios-cache-timeout | netbios-explorer-timeout | netbios-group-cache | netbios-retry-interval | netbios-verify-interval | sna-cache-timeout | explorer-delay-time | sna-explorer-timeout | explorer-wait-time | sna-group-cache | sna-retry-interval | sna-verify-interval} time`	Tunes an existing configuration parameter.

NetBIOS Dial-on-Demand Routing

This feature allows you to transport NetBIOS in a dial-on-demand routing (DDR) environment by filtering NetBIOS Session Alive packets from the WAN. NetBIOS periodically sends Session Alive packets as LLC2 I-frames. These packets do not require a response and are superfluous to the function of proper data flow. Furthermore, these packets keep dial-on-demand interfaces up and this up time causes unwanted per-packet charges in DDR networks. By filtering these NetBIOS Session Alive packets, you reduce traffic on the WAN and you reduce some costs that are associated with dial-on-demand routing.

To enable NetBIOS DDR, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw netbios keepalive-filter`	Enables NetBIOS DDR.

The following command enables NetBIOS DDR:

```
dlsw netbios keepalive-filter
```

SNA Dial-on-Demand Routing

This feature allows you to run DLSw+ over a switched line and have the Cisco IOS software take the switched line down dynamically when it is not in use. Utilizing this feature gives the IP Routing table more time to converge when a network problem hinders a remote peer connection. In small networks with good IP convergence time and ISDN lines that start quickly, it is not as necessary to use the **keepalive** option. To use this feature, you must set the **keepalive** value to zero, and you may need to use a lower value for the **timeout** option than the default, which is 90 seconds.

To configure SNA DDR, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **tcp** *ip-address* [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cluster** *cluster-id*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**inactivity**] [**dynamic**] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**] [**rif-passthru** *virtual-ring-number*] [**tcp-queue-max** *size*] [**timeout** *seconds*]	Configures SNA DDR.

The following command configures the SNA DDR feature:

```
dlsw remote-peer 0 tcp 10.2.13.4 keepalive 0
```

UDP Unicast Feature

The UDP Unicast feature sends the SSP address resolution packets via UDP unicast service rather than TCP. (SSP packets include: CANUREACH_EX, NETBIOS_NQ_ex, NETBIOS_ANQ, and DATAFRAME.) The UDP unicast feature allows DLSw+ to better control address resolution packets and unnumbered information frames during periods of congestion. Previously, these frames were carried over TCP. TCP resends frames that get lost or delayed in transit, and hence aggravate congestion. Because address resolution packets and unnumbered information frames are not sent on a reliable transport on the LAN, sending them reliably over the WAN is unnecessary. By using UDP for these frames, DLSw+ minimizes network congestion.

Note UDP unicast enhancement has no affect on DLSw+ FST or direct peer encapsulation.

This feature is enabled by default. To disable User Datagram Protocol (UDP) Unicast, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw udp-disable**	Disables UDP Unicast.

LLC1 Circuits

Support for LLC1 circuits more efficiently transports LLC1 UI traffic across a DLSw+ cloud. With LLC1 circuit support, the LLC1 unnumbered information frames (UI) are no longer subject to input queueing and are guaranteed to traverse the same path for the duration of the flow. This feature improves transportation of LLC1 UI traffic because there is no longer the chance of having a specifically routed LLC1 UI frame broadcast to all remote peers. The circuit establishment process has not changed except that the circuit is established as soon as the specifically routed LLC1 UI frame is received and the DLSw+ knows of reachability for the destination MAC address. Furthermore, the connection remains in the CIRCUIT_ESTABLISHED state (rather than proceeding to the CONNECT state) until there is no UI frame flow for a MAC/SAP pair for 10 minutes.

This feature is enabled by default.

Dynamic Peers

In TCP encapsulation, the **dynamic** option and its suboptions **no-llc** and **inactivity** allow you to specify and control the activation of dynamic peers, which are configured peers that are activated only when required. Dynamic peer connections are established only when there is DLSw+ data to send. The dynamic peer connections are taken down when the last LLC2 connection using them terminates and the time period specified in the **no-llc** option expires. You can also use the **inactivity** option to take down dynamic peers when the circuits using them are inactive for a specified number of minutes.

Note Because the **inactivity** option may cause active LLC2 sessions to be terminated, you should not use this option unless you want active LLC2 sessions to be terminated.

To configure a dynamic peer, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw remote-peer** *list-number* **tcp ip-address** [**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**circuit-weight** *weight*] [**cluster** *cluster-id*] [**cost** *cost*] [**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**inactivity**] [**dynamic**] [**keepalive** *seconds*] [**lf** *size*] [**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**] [**rif-passthru** *virtual-ring-number*] [**tcp-queue-max** *size*] [**timeout** *seconds*]	Configures a dynamic peer.

The following command specifies a dynamic peer with TCP encapsulation:

```
dlsw remote-peer 0 tcp 10.23.4.5 dynamic
```

Promiscuous Peer Defaults

If you do not configure a **dlsw remote-peer** statement on the DLSw+ router, then you must specify the **promiscuous** keyword on the **dlsw local-peer** statement. The **promiscuous** keyword enables the router to accept peer connection requests from those routers that are not preconfigured. Setting the **dlsw prom-peer-defaults** command allows the user to determine various settings for the promiscuous transport.

To configure promiscuous peer defaults, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw prom-peer-defaults** [**bytes-netbios-out** *bytes-list-name*] [**cost** *cost*] [**dest-mac** *destination-mac-address*] [**dmac-output-list** *access-list-number*] [**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*] [**lsap-output-list** *list*] [**tcp-queue-max** *size*]	Configures promiscuous peer defaults.

Availability

DLSw+ supports the following features that allow it to dynamically finds alternate paths quickly and optionally load balances across multiple active peers, ports, and channel gateways:

- Load Balancing, page 23
- Ethernet Redundancy, page 25
- Backup Peers, page 25

Load Balancing

DLSw+ offers enhanced availability by caching multiple paths to a given MAC address or NetBIOS name (where a path is either a remote peer or a local port). Maintaining multiple paths per destination is especially attractive in SNA networks. A common technique used in the hierarchical SNA environment is assigning the same MAC address to different Token Ring interface couplers (TICs) on the IBM FEPs. DLSw+ ensures that duplicate TIC addresses are found, and, if multiple DLSw+ peers can be used to reach the FEPs, they are cached.

The way that multiple capable peers are handled with DLSw+ can be configured to meet either of the following network needs:

- Fault tolerance—To rapidly reconnect if a data-link connection is lost. If load balancing is not enabled, the Cisco IOS software, by default, maintains a preferred path and one or more capable paths to each destination. The preferred path is either the peer or port that responds first to an explorer frame or the peer with the least cost. If the preferred path to a given destination is unavailable, the next available capable path is promoted to the new preferred path. No additional broadcasts are required, and recovery through an alternate peer is immediate. Maintaining multiple cache entries facilitates a timely reconnection after session outages.

  A peer with the least cost can also be the preferred path. You can specify cost in either the **dlsw local peer** or **dlsw remote peer** commands. See the *DLSw+ Design and Implementation Guide* for details on how cost can be applied to control which path sessions use.

- Load balancing—To distribute the network traffic over multiple DLSw+ peers in the network. Alternately, when there are duplicate paths to the destination end system, you can configure load balancing. DLSw+ alternates new circuit requests in either a round-robin or *enhanced* load balancing fashion through the list of capable peers or ports. If round-robin is configured, the router distributes the new circuit in a round-robin fashion, basing it's decision on which peer or port established the last circuit. If enhanced load balancing is configured, the router distributes new circuits based on existing loads and the desired ratio. It detects the path that is underloaded in comparison to the other capable peers and will assign new circuits to that path until the desired ratio is achieved.

For multiple peer connections, peer costs must be applied. The DLSw+ Enhanced Load Balancing feature works only with the lowest (or equal) cost peers. For example, if the user specifies dlswrtr1, dlswrtr2 and dlswrtr3 with costs of 4, 3, and 3 respectively, DLSw+ establishes new circuits with only dlswrtr 2 and dlswrtr3.

To enable the DLSw+ Enhanced Load Balancing feature on the local router, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw load-balance [round-robin | circuit count circuit-weight]`	Configures the DLSw+ Enhanced Load Balancing feature on the local router.

To adjust the circuit weight for a remote peer with TCP encapsulation, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw remote-peer tcp [circuit-weight value]`	Adjusts the circuit weight on the remote peer.

To adjust the circuit weight for a remote peer with DLSw+ Lite encapsulation, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw remote-peer frame-relay interface serial number dlci number [circuit-weight value]`	Adjusts the circuit weight on the remote peer.

The circuit-weight of a remote peer controls the number of circuits that peer can take. If multiple, equally low-cost peers can reach a remote source, the circuits to that remote source are distributed among the remote peers based on the ratio of their configured circuit-weights. The peer with the highest circuit-weight takes more circuits.

Because a DLSw+ peer selects its new circuit paths from within its reachability cache, the user must configure the **dlsw timer explorer-wait-time** command with enough time to allow for all the explorer responses to be received. If the new DLSw+ Enhanced Load Balancing Feature is enabled, a message is displayed on the console to alert the user if the timer is not set.

To configure the amount of time needed for all the explorer responses to be received, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw timer {explorer-wait-time}`	Sets the time to wait for all stations to respond to explorers.

See the *DLSw+ Design and Implementation Guide* for details on how to configure load balancing in DLSw+. Refer to the "DLSw+ with Enhanced Load Balancing Configuration Example" section on page 47 for a sample configuration.

Ethernet Redundancy

The DLSw+ Ethernet Redundancy feature, introduced in Cisco IOS Release 12.0(5)T, provides redundancy and load balancing between multiple DLSw+ peers in an Ethernet environment. It enables DLSw+ to support parallel paths between two points in an Ethernet environment, ensuring resiliency in the case of a router failure and providing load balancing for traffic load. The feature also enables DLSw+ to support multiple DLSw+ routers on the same transparent bridged domain that can reach the same MAC address in a switched environment.

To enable the DLSw+ Ethernet Redundancy feature, issue the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dlsw transparent redundancy-enable`	Configures transparent redundancy.

To enable the DLSw+ Ethernet Redundancy feature in a switched environment, enter the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# dlsw transparent switch-support`	Enables DLSw+ Ethernet Redundancy feature when using a switch device.
Step 2	`Router(config-if)# dlsw transparent map local mac` *mac address* `remote mac` *mac address* `neighbor` *mac address*	Configures a single destination MAC address to which multiple MAC addresses on a transparent bridged are mapped.

The Ethernet Redundancy feature is a complex feature. See the *DLSw+ Design and Implementation Guide* for more details. Refer to the "DLSw+ with Ethernet Redundancy Configuration Example" section on page 51 and the "DLSw+ with Ethernet Redundancy Enabled for Switch Support Configuration Example" section on page 52 for sample configurations.

Backup Peers

The **backup-peer** option is common to all encapsulation types on a remote peer and specifies that this remote peer is a backup peer for the router with the specified IP-address, Frame Relay Data-Link Control Identifier (DLCI) number, or interface name. When the primary peer fails, all circuits over this peer are disconnected and the user can start a new session via their backup peer. Prior to Cisco IOS Release 11.2(6)F, you could configure backup peers only for primary FST and TCP.

Also, when you specify the **backup-peer** option in a **dlsw remote-peer tcp** command, the backup peer is activated only when the primary peer becomes unreachable. Once the primary peer is reactivated, all new sessions use the primary peer and the backup peer remains active only as long as there are LLC2 connections using it. You can use the **linger** option to specify a period (in minutes) that the backup peer remains connected after the connection to the primary peer is reestablished. When the linger period expires, the backup peer connection is taken down.

Note If the **linger** keyword is set to 0, all existing sessions on the backup router immediately drop when the primary recovers. If the **linger** keyword is omitted, all existing sessions on the backup router remain active (as long as the session is active) when the primary recovers, however, all new sessions establish via the primary peer. If the **linger** keyword is set to *x* minutes, all existing sessions on the backup router remain active for *x* minutes once the primary recovers, however, all new sessions establish via the primary peer. Once *x* minutes expire, all existing sessions on the backup router drop and the backup peer connection is terminated. The **linger** keyword can be used to minimize line costs if the backup peer is accessed over dial lines, but can be set high enough to allow an operator warning to be sent to all the SNA end users. It will not, however, pass explorers and will not create any new circuits while the primary is up.

To configure a backup peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw remote peer backup-peer` *ip-address*	Configures a backup peer.

Modes of Operation

It is sometimes necessary for DLSw+ and RSRB to coexist in the same network and in the same router (for example, during migration from RSRB to DLSw+). Cisco DLSw+ supports this environment. In addition, DLSw+ must also interoperate with other vendors' implementations that are based upon other DLSw RFC standards, such as DLSw Version 1 and Version 2.

Cisco routers, implementing Cisco DLSw+, automatically supports three different modes of operation:

- Dual mode—A Cisco router can communicate with some remote peers using RSRB and with others using DLSw+, providing a smooth migration path from RSRB to DLSw+; in dual mode, RSRB and DLSw+ coexist on the same box; the local peer must be configured for both RSRB and DLSw+; and the remote peers must be configured for either RSRB or DLSw, but not both.
- Standards compliance mode—DLSw+ can detect automatically (via the DLSw capabilities exchange) if the participating router is manufactured by another vendor, therefore operating in DLSw standard mode (DLSw Version 1 RFC 1795 and DLSw Version 2 RFC 2166).
- Enhanced mode—DLSw+ can detect automatically that the participating router is another DLSw+ router, therefore operating in enhanced mode, making all of the features of DLSw+ available to the SNA and NetBIOS end systems.

Note DLSw+ does not interoperate with the DLSw RFC 1434 standard.

Some enhanced DLSw+ features are also available when a Cisco router is operating in standards compliance mode with another vendor's router. In particular, enhancements that are locally controlled options on a router can be accessed even though the remote router does not have DLSw+. These include reachability caching, explorer firewalls and media conversion.

Network Management

There are several network management tools available to the user to help them more easily manage and troubleshoot their DLSw+ network. CiscoWorks Blue Maps provides a logical view of the portion of your router network relevant to DLSw+ (there is a similar tool for RSRB and APPN). CiscoWorks Blue SNA View adds to the information provided by Maps by correlating SNA PU and LU names with DLSw+ circuits and DLSw+ peers. CiscoWorks Blue Internetwork Status Monitor (ISM) support allows you to manage your router network from the mainframe console using IBM's NetView or Sterling's SOLVE:Netmaster. See the *DLSw+ Design and Implementation Guide* "Using CiscoWorks Blue: Maps, SNA View, and Internetwork Status Monitor" chapter for more details.

Traffic Bandwidth and Queueing Management

Cisco offers several bandwidth management and queueing features (such as DLSw+ RSVP) to enhance the overall performance of your DLSw+ network. The queueing and bandwidth management features are described in detail in the *DLSw Design and Implementation Guide* "Bandwidth Management Queueing" chapter.

Access Control

DLSw+ offers the following features that allow it to control access to various resources throughout a network:

DLSw+ Ring List or Port List

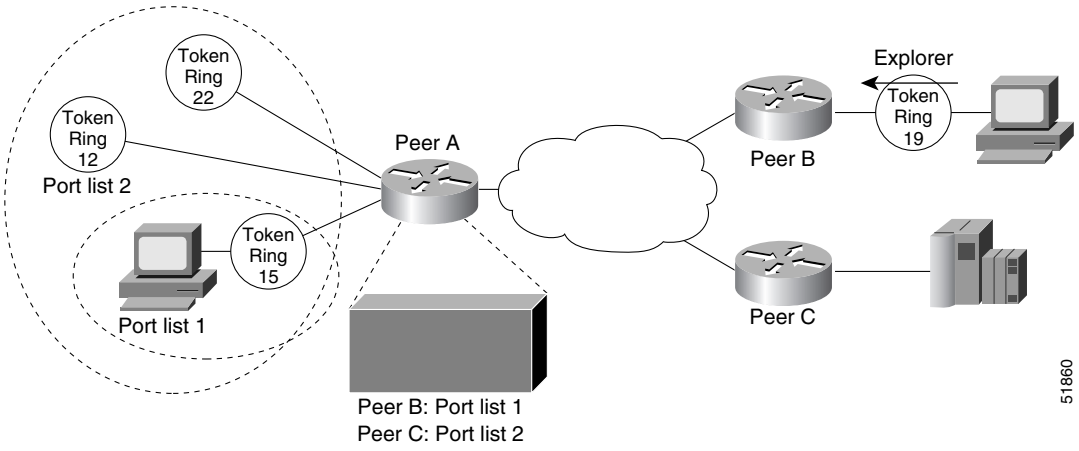
DLSw+ ring lists map traffic on specific local rings to remote peers. You can create a ring list of local ring numbers and apply the list to remote peer definitions. Traffic received from a remote peer is only forwarded to the rings specified in the ring list. Traffic received from a local interface is only forwarded to peers if the input ring number appears in the ring list applied to the remote peer definition. The definition of a ring list is optional. If you want all peers and all rings to receive all traffic, you do not have to define a ring list. Simply specify 0 for the list number in the remote peer statement.

To define a ring list, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw ring-list` *list-number* `rings` *ring-number*	Defines a ring list.

DLSw+ port lists map traffic on a local interface (either Token Ring or serial) to remote peers. Port lists do not work with Ethernet interfaces, or any other interface types connected to DLSw+ by means of a bridge group. You can create a port list of local ports and apply the list to remote peer definitions. Traffic received from a remote peer is only forwarded to peers if the input port number appears in the port list applied to the remote peer definition. The port list command provides a single command to specify both serial and Token Ring interfaces. Figure 130 shows how port lists are used to map traffic.

Figure 130 Mapping Traffic Using Port Lists



The definition of a port list is optional. If you want all peers and all interfaces to receive all traffic, you do not have to define a port list. Simply specify 0 for the list number in the remote peer statement.

To define a port list, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw port-list list-number type number`	Defines a port list.

Note Either the ring list or the port list command can be used to associate rings with a given ring list. The ring list command is easier to type in if you have a large number of rings to define.

DLSw+ Bridge Group List

DLSw+ bridge group lists map traffic on the local Ethernet bridge group interface to remote peers. You can create a bridge group list and apply the list to remote peer definitions. Traffic received from a remote peer is only forwarded to the bridge group specified in the bridge group list. Traffic received from a local interface is only forwarded to peers if the input bridge group number appears in the bridge group list applied to the remote peer definition. The definition of a bridge group list is optional. Because each remote peer has a single list number associated with it, if you want traffic to go to a bridge group and to either a ring list or port list, you should specify the same list number in each definition

To define a bridge-group list, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw bgroup-list list-number bgroups number`	Defines a ring list.

Static Paths

Static path definitions allow a router to setup circuits without sending explorers. The path specifies the peer to use to access a MAC address or NetBIOS name.

To configure static paths to minimize explorer traffic originating in this peer, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# dlsw mac-addr mac-addr {ring ring number | remote-peer {interface serial number | ip-address ip-address} | rif rif string | group group}`	Configures the location or path of a static MAC address.
or	or
`Router(config)# dlsw netbios-name netbios-name {ring ring number | remote-peer {interface serial number | ip-address ip-address} | rif rif string | group group}`	Configures a static NetBIOS name.

Static Resources Capabilities Exchange

To reduce explorer traffic destined for this peer, the peer can send other peers a list of resources for which it has information (**icanreach**) or does not have information (**icannotreach**). This information is exchanged as part of a capabilities exchange.To configure static resources that will be exchanged as part of a capabilities exchange, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# dlsw icannotreach saps sap [sap...]`	Configures a resource not locally reachable by the router.
or	or
`Router(config)# dlsw icanreach {mac-exclusive | netbios-exclusive | mac-address mac-addr [mask mask] | netbios-name name | saps}`	Configures a resource locally reachable by the router.

Filter Lists in the Remote-Peer Command

The **dest-mac** and **dmac-output-list** options allow you to specify filter lists as part of the **dlsw remote-peer** command to control access to remote peers. For static peers in direct, FST, or TCP encapsulation, these filters control which explorers are sent to remote peers. For dynamic peers in TCP encapsulation, these filters also control the activation of the dynamic peer. For example, you can specify at a branch office that a remote peer is activated only when there is an explorer frame destined for the Media Access Control (MAC) address of an FEP.

The **dest-mac** option permits the connection to be established only when there is an explorer frame destined for the specified MAC address. The **dmac-output-list** option permits the connection to be established only when the explorer frame passes the specified access list. To permit access to a single MAC address, use the **dest-mac** option, because it is a configuration “shortcut” compared to the **dmac-output-list** option.

Configuring DLSw+ Timers

To configure DLSw+ timers, use the following command in global configuration mode:

Command	Purpose
Router(config)# **dlsw timer** {**icannotreach-block-time** | **netbios-cache-timeout** | **netbios-explorer-timeout** | **netbios-group-cache** |**netbios-retry-interval** | **netbios-verify-interval** |**sna-cache-timeout** | **sna-explorer-timeout** | **sna-group-cache** | **sna-retry-interval** | **sna-verify-interval**} *time*	Configures DLSw+ timers.

See the *DLSw+ Design and Implementation Guide* "Customization" chapter and the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2) for command details.

Verifying DLSw+

To verify that DLSw+ is configured on the router, use the following command in privileged EXEC mode:

Command	Purpose
Router# **show dlsw capabilities local**	Displays the DLSw+ configuration of a specific peer.

The following sample shows that DLSw+ is configured on router milan:

```
milan#show dlsw capabilities local
DLSw:Capabilities for peer 1.1.1.6(2065)
vendor id (OUI)         :'00C' (cisco)
  version number          :1
  release number          :0
  init pacing window      :20
  unsupported saps        :none
  num of tcp sessions     :1
  loop prevent support    :no
  icanreach mac-exclusive :no
  icanreach netbios-excl. :no
  reachable mac addresses :none
  reachable netbios names :none
  cisco version number    :1
  peer group number       :0
  border peer capable     :no
  peer cost               :3
  biu-segment configured  :no
  UDP Unicast support     :yes
  local-ack configured    :yes
  priority configured     :no
Cisco Internetwork Operating System Software IOS GS Software (GS7-K-M),
Experimental Version 11.1(10956) [sbales 139]
Copyright (c) 1986-1996 by cisco Systems, Inc.
Compiled Thu 30-May-96 09:12 by sbales8
```

If only a command prompt appears, then DLSw+ is not configured for the router.

Alternately, to verify that DLSw+ is configured, issue the following command in privileged EXEC mode:

Command	Purpose
Router# **show running configuration**	Displays the running configuration of a device.

The global DLSw+ configuration statements, including the **dlsw local-peer** statement, appear in the output before the interface configuration statements. The following sample shows that DLSw+ is configured on router milan:

```
milan# show run
version 12.0
!
hostname Sample
!
source-bridge ring-group 110
dlsw local-peer peer-id 10.1.1.1 promiscuous
!
interface TokenRing0/0
no ip address
ring-speed 16
source-bridge 222 1 110
source-bridge spanning
!
```

Monitoring and Maintaining the DLSw+ Network

To monitor and maintain activity on the DLSw+ network, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
Router# **show dlsw capabilities interface** *type number*	Displays capabilities of a direct-encapsulated remote peer.
Router# **show dlsw capabilities ip-address** *ip-address*	Displays capabilities of a TCP/FST remote peer.
Router# **show dlsw capabilities local**	Displays capabilities of the local peer.
Router# **show dlsw circuits**	Displays DLSw+ circuit information.
Router# **show dlsw fastcache**	Displays the fast cache for FST and direct-encapsulated peers.
Router# **show dlsw local-circuit**	Displays DLSw+ circuit information when doing local conversion.
Router# **show dlsw peers**	Displays DLSw+ peer information.
Router# **show dlsw reachability**	Displays DLSw+ reachability information.
Router# **dlsw disable**	Disables and re-enable DLSw+ without altering the configuration.
Router# **show dlsw statistics** [**border-peers**]	Displays the number of frames that have been processed in the local, remote, and group caches.
Router# **clear dlsw circuit**	Closes all the DLSw+ circuits[1]. Also used to reset to zero the number of frames that have been processed in the local, remote, and group cache.

1. Issuing the **clear dlsw circuits** command will cause the loss of any associated LLC2 sessions.

See the *DLSw+ Design and Implementation Guide* "Using Show and Debug Commands" chapter and the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2) for details of the commands.

DLSw+ Configuration Examples

The following sections provide DLSw+ configuration examples:

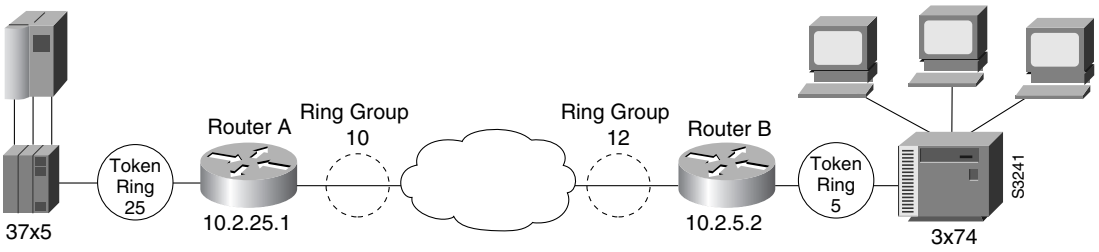
DLSw+ Using TCP Encapsulation and LLC2 Local Acknowledgment—Basic Configuration Example

This sample configuration requires the following tasks, which are described in earlier sections of this document:

- Define a Source-Bridge Ring Group for DLSw+
- Define a DLSw+ Local Peer for the Router
- Define DLSw+ Remote Peers
- Assign DLSw+ to a local data-link control

Figure 131 illustrates a DLSw+ configuration with local acknowledgment. Because the RIF is terminated, the ring group numbers do not have to be the same.

Figure 131 DLSw+ with Local Acknowledgment—Simple Configuration



Router A

```
source-bridge ring-group 10
!
dlsw local-peer peer-id 10.2.25.1
dlsw remote-peer 0 tcp 10.2.5.2
 interface loopback 0
 ip address 10.2.25.1 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 25 1 10
 source-bridge spanning
```

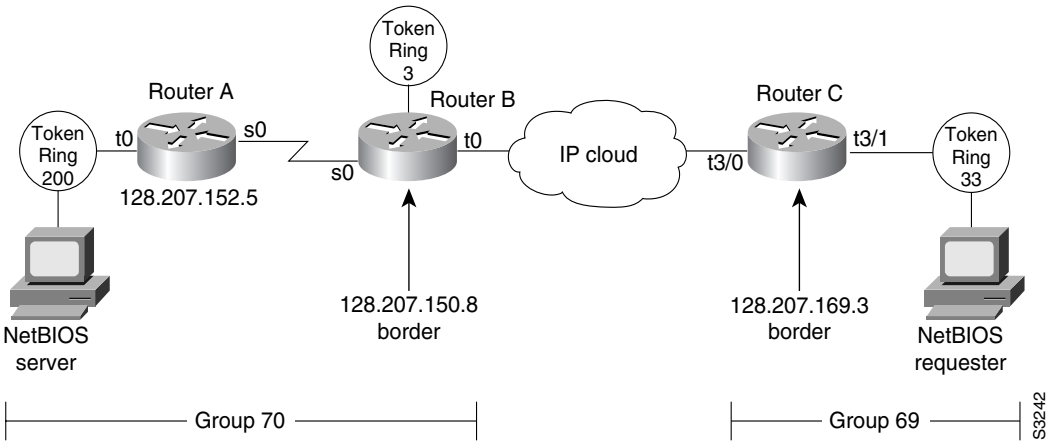
Router B

```
source-bridge ring-group 12
dlsw local-peer peer-id 10.2.5.2
dlsw remote-peer 0 tcp 10.2.25.1
 interface loopback 0
 ip address 10.2.5.2 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 5 1 12
 source-bridge spanning
```

DLSw+ Using TCP Encapsulation with Local Acknowledgment—Peer Group Configuration Example 1

Figure 132 illustrates border peers with TCP encapsulation. Router A is configured to operate in promiscuous mode, and border peers Routers B and C forward broadcasts. This configuration reduces processing requirements in Router A (the access router) and still supports any-to-any networks. Configure Border peer B and C so that they peer to each other.

Figure 132 DLSw+ with Peer Groups Specified (Example 1)



Router A

```
hostname Router A
!
source-bridge ring group 31
dlsw local-peer peer-id 128.207.152.5 group 70 promiscuous
dlsw remote peer 0 tcp 128.207.150.8
interface loopback 0
ip address 128.207.152.5 255.255.255.0
!
interface serial 0
 ip unnumbered tokenring
 clockrate 56000
!
interface tokenring 0
 ip address 128.207.152.5 255.255.255.0
 ring-speed 16
 source-bridge 200 13 31
 source-bridge spanning
!
router igrp 777
network 128.207.0.0
```

Router B

```
hostname Router B
!
source-bridge ring-group 31
dlsw local-peer peer-id 128.207.150.8 group 70 border promiscuous
dlsw remote-peer 0 tcp 128.207.169.3
interface loopback 0
ip address 128.207.150.8 255.255.255.0
```

```
!
interface serial 0
 ip unnumbered tokenring 0
 bandwidth 56
!
interface tokenring 0
 ip address 128.207.150.8 255.255.255.0
 ring-speed 16
source-bridge 3 14 31
 source-bridge spanning
!
router igrp 777
network 128.207.0.0
```

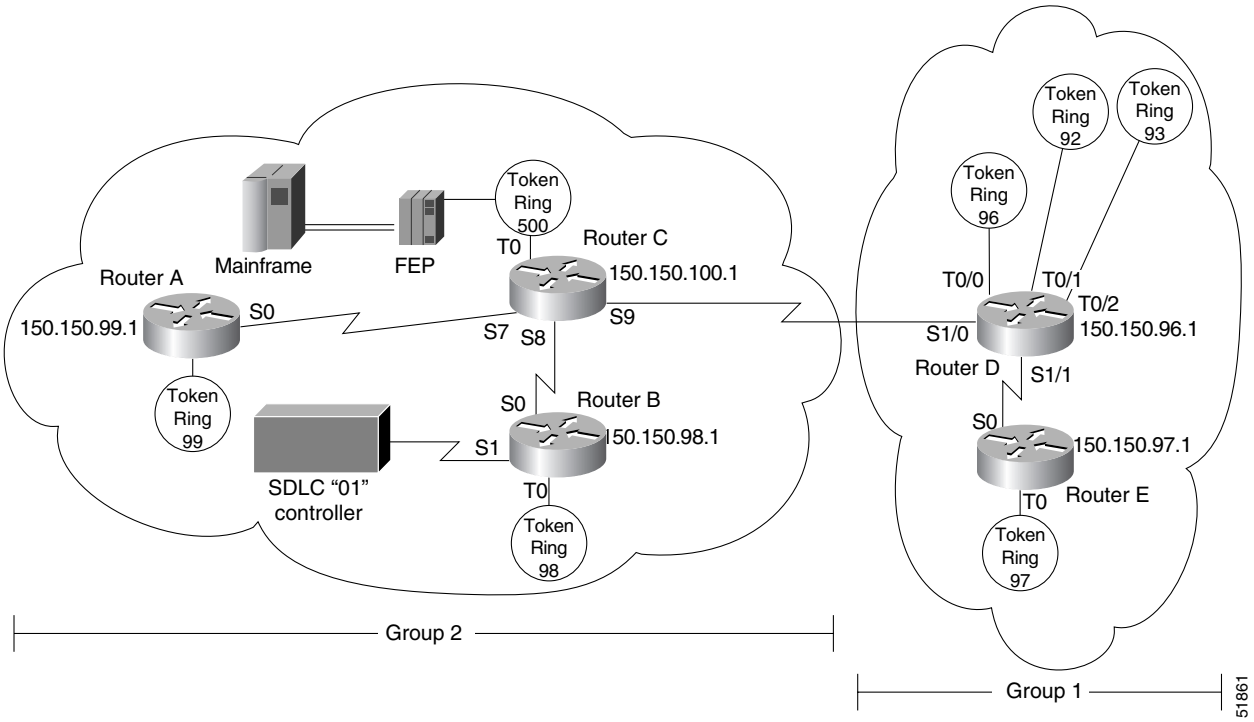
Router C

```
hostname Router C
!
source-bridge ring-group 69
dlsw local-peer peer-id 128.207.169.3 group 69 border promiscuous
dlsw remote-peer 0 tcp 128.207.150.8
interface loopback 0
ip address 128.207.169.3 255.255.255.0
!
interface tokenring 3/0
description fixed to flashnet
 ip address 128.207.2.152 255.255.255.0
 ring-speed 16
 multiring all
!
interface tokenring 3/1
 ip address 128.207.169.3 255.255.255.0
 ring-speed 16
source-bridge 33 2 69
 source-bridge spanning
!
router igrp 777
network 128.207.0.0
```

DLSw+ Using TCP Encapsulation with Local Acknowledgment—Peer Group Configuration Example 2

Figure 133 illustrates a peer group configuration that allows any-to-any connection except for Router B. Router B has no connectivity to anything except router C because the **promiscuous** keyword is omitted.

Figure 133 DLSw+ with Peer Groups Specified (Example 2)



Router A

```
hostname Router A
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.99.1 group 2 promiscuous
dlsw remote-peer 0 tcp 150.150.100.1
!
interface loopback 0
 ip address 150.150.99.1 255.255.255.192
!
interface tokenring 0
 no ip address
 ring-speed 16
source-bridge 99 1 2000
 source-bridge spanning
!
router eigrp 202
network 150.150.0.0
```

Router B

```
hostname Router B
!
source-bridge ring-group 2000
```

```
dlsw local-peer peer-id 150.150.98.1 group 2
dlsw remote-peer 0 tcp 150.150.100.1
!
interface loopback 0
 ip address 150.150.98.1 255.255.255.192
!
interface serial 1
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 9600
 sdlc role primary
 sdlc vmac 4000.8888.0100
 sdlc address 01
 sdlc xid 01 05d20006
 sdlc partner 4000.1020.1000 01
 sdlc dlsw 1
!
interface tokenring 0
 no ip address
 ring-speed 16
source-bridge 98 1 2000
 source-bridge spanning
!
router eigrp 202
network 150.150.0.0
```

Router C

```
hostname Router C
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.100.1 group 2 border promiscuous
dlsw remote-peer 0 tcp 150.150.96.1
!
interface loopback 0
 ip address 150.150.100.1 255.255.255.192
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 500 1 2000
 source-bridge spanning
!
router eigrp 202
network 150.150.0.0
```

Router D

```
hostname Router D
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.96.1 group 1 border promiscuous
dlsw remote-peer 0 tcp 150.150.100.1
!
interface loopback 0
 ip address 150.150.96.1 255.255.255.192
!
interface tokenring 0/0
 no ip address
 ring-speed 16
 source-bridge 96 1 2000
 source-bridge spanning
!
interface tokenring 0/1
```

```
 no ip address
 ring-speed 16
 source-bridge 92 1 2000
 source-bridge spanning
!
.interface tokenring 0/2
 no ip address
 ring-speed 16
 source-bridge 93 1 2000
 source-bridge spanning
!
router eigrp 202
network 150.150.0.0
```

Router E

```
hostname Router E
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.97.1 group 1 promiscuous
dlsw remote-peer 0 tcp 150.150.96.1
!
interface loopback 0
 ip address 150.150.97.1 255.255.255.192
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 97 1 2000
 source-bridge spanning
!
router eigrp 202
network 150.150.0.0
```

DLSw+ with SDLC Multidrop Support Configuration Examples

In the following example, all devices are type PU 2.0:

```
interface serial 2
 mtu 4400
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 19200
 sdlc role primary
 sdlc vmac 4000.1234.5600
 sdlc address C1
 sdlc xid C1 05DCCCC1
 sdlc partner 4001.3745.1088 C1
 sdlc address C2
 sdlc xid C2 05DCCCC2
 sdlc partner 4001.3745.1088 C2
 sdlc dlsw C1 C2
```

The following example shows mixed PU 2.0 (device using address C1) and PU 2.1 (device using address C2) devices:

```
interface serial 2
 mtu 4400
 no ip address
 encapsulation sdlc
 no keepalive
```

```
 clockrate 19200
 sdlc role primary
 sdlc vmac 4000.1234.5600
 sdlc address C1
 sdlc xid C1 05DCCCC1
 sdlc partner 4001.3745.1088 C1
 sdlc address C2 xid-poll
 sdlc partner 4001.3745.1088 C2
 sdlc dlsw C1 C2
```

In the following example, all devices are type PU 2.1 (Method 1):

```
interface serial 2
 mtu 4400
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 19200
 sdlc role primary
 sdlc vmac 4000.1234.5600
 sdlc address C1 xid-poll
 sdlc partner 4001.3745.1088 C1
 sdlc address C2 xid-poll
 sdlc partner 4001.3745.1088 C2
 sdlc dlsw C1 C2
```

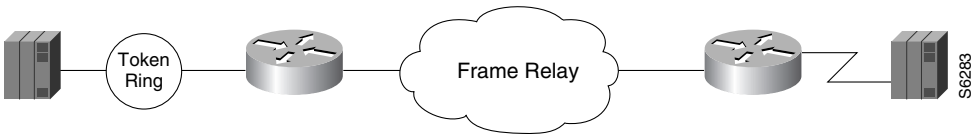
In the following example, all devices are type PU 2.1 (Method 2):

```
interface serial 2
 mtu 4400
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 19200
 sdlc role prim-xid-poll
 sdlc vmac 4000.1234.5600
 sdlc address C1
 sdlc partner 4001.3745.1088 C1
 sdlc address C2
 sdlc partner 4001.3745.1088 C2
 sdlc dlsw C1 C2
```

DLSw+ with LLC2-to-SDLC Conversion Between PU 4-to-PU 4 Communication Example

The following example is a sample configuration for LLC2-to-SDLC conversion for PU 4-to-PU 4 communication as shown in Figure 134:

Figure 134 LLC2-to-SDLC Conversion for PU 4-to-PU 4 Communication



Router A

```
source-bridge ring-group 1111
```

```
dlsw local-peer peer-id 10.2.2.2
dlsw remote-peer 0 tcp 10.1.1.1
interface loopback 0
ip address 10.2.2.2 255.255.255.0
interface TokenRing 0
 no ip address
 ring-speed 16
source-bridge 2 1111
source-bridge spanning
```

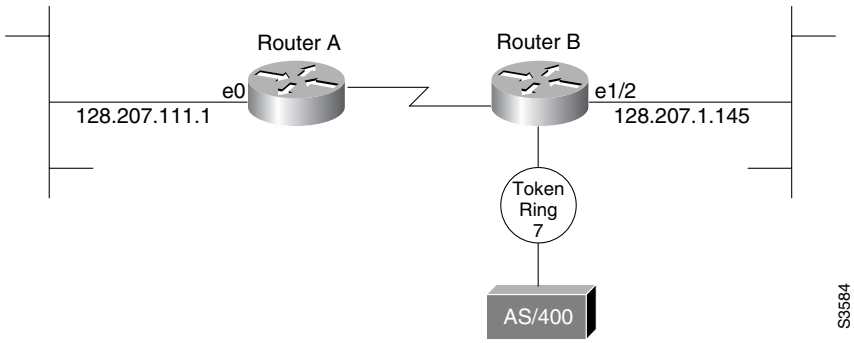
Router B

```
dlsw local-peer peer-id 10.1.1.1
dlsw remote-peer 0 tcp 10.2.2.2
interface loopback 0
ip address 10.1.1.1 255.255.255.0
interface serial 0
 mtu 4096
 no ip address
 encapsulation sdlc
 no keepalive
 nzri-encoding
 clockrate 9600
 sdlc vmac 4000.3745.0000
 sdlc N1 48016
 sdlc address 04 echo
 sdlc partner 4000.1111.0020 04
 sdlc dlsw 4
```

DLSw+ Translation Between Ethernet and Token Ring Configuration Example

DLSw+ also supports Ethernet media. The configuration is similar to other DLSw+ configurations, except for configuring for a specific media. The following example shows Ethernet media (see Figure 135).

Figure 135 DLSw+ Translation Between Ethernet and Token Ring



Router A

```
hostname Router A
!
dlsw local-peer peer-id 128.207.111.1
dlsw remote-peer 0 tcp 128.207.1.145
dlsw bridge-group 5
!
```

```
interface loopback 0
ip address 128.207.111.1 255.255.255.0
interface Ethernet 0
no ip address
 bridge-group 5
!
bridge 5 protocol ieee
```

Router B

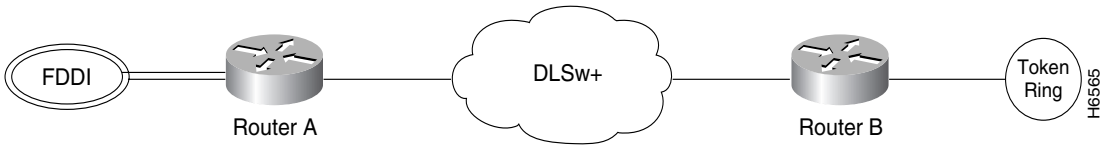
```
hostname Router B
!
source-bridge transparent 500 1000 1 5
dlsw local-peer peer-id 128.207.1.145
dlsw remote-peer 0 tcp 128.207.111.1
dlsw bridge-group 5
!
interface loopback 0
ip address 128.207.1.145 255.255.255.0
interface ethernet 1/2
no ip address
 bridge-group 5
!
interface tokenring 2/0
 no ip address
 ring-speed 16
 source-bridge 7 1 500
 source-bridge spanning
!
bridge 5 protocol ieee
```

Because DLSw+ does not do local translation between different LAN types, Router B must be configured for SR/TLB by issuing the **source-bridge transparent** command. Also, note that the bridge groups are configured on the ethernet interfaces.

DLSw+ Translation Between FDDI and Token Ring Configuration Example

DLSw+ also supports FDDI media. The configuration is similar to other DLSw+ configurations except for configuring for a specific media type. The following example shows FDDI media (see Figure 136).

Figure 136 DLSw+ Translation Between FDDI and Token Ring



In the following configuration, an FDDI ring on Router A is connected to a Token Ring on Router B across a DLSw+ link.

Router A

```
source-bridge ring-group 10
dlsw local-peer peer-id 132.11.11.2
dlsw remote-peer 0 tcp 132.11.11.3
interface loopback 0
ip address 132.11.11.2 255.255.255.0
interface fddi 0
 no ip address
```

```
 source-bridge 26 1 10
 source-bridge spanning
```

Router B

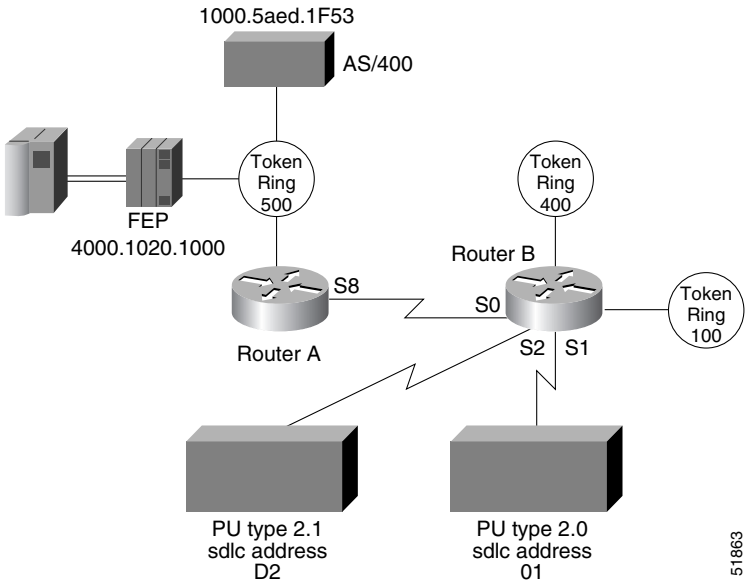
```
source-bridge ring-group 10
dlsw local peer peer-id 132.11.11.3
dlsw remote-peer 0 tcp 132.11.11.2
interface loopback 0
ip address 132.11.11.3 255.255.255.0
interface tokenring 0
 no ip address
 source-bridge 25 1 10
 source-bridge spanning
```

DLSw+ Translation Between SDLC and Token Ring Media Example

DLSw+ provides media conversion between local or remote LANs and SDLC. For additional information about configuring SDLC parameters, refer to the chapter "Configuring LLC2 and SDLC Parameters."

Figure 137 illustrates DLSw+ with SDLC encapsulation. For this example, 4000.1020.1000 is the MAC address of the FEP host (PU 4.0). The MAC address of the AS/400 host is 1000.5aed.1f53, which is defined as Node Type 2.1. Router B serves as the primary station for the remote secondary station 01. Router B can serve as either primary station or secondary station to remote station D2.

Figure 137 DLSw+ Translation Between SDLC and Token Ring Media



Router A

```
hostname Router A
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.10.2
dlsw remote-peer 0 tcp 150.150.10.1
!
interface loopback 0
```

```
ip address 150.150.10.2 255.255.255.0
interface serial 8
 ip address 150.150.11.2 255.255.255.192
 clockrate 56000

!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 500 1 2000
 source-bridge spanning
!
router eigrp 202
 network 150.150.0.0
```

Router B

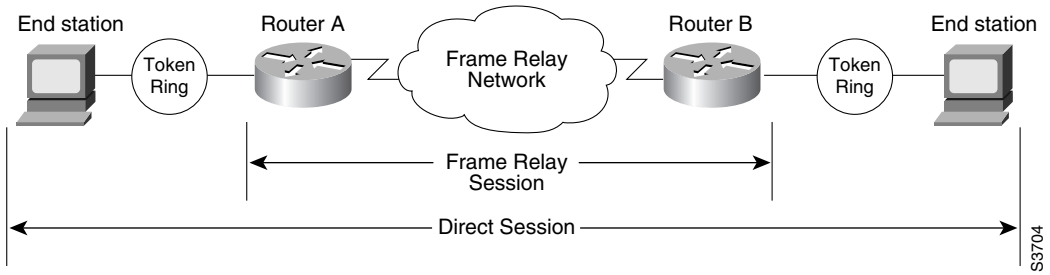
```
hostname Router B
!
source-bridge ring-group 2000
dlsw local-peer peer-id 150.150.10.1
dlsw remote-peer 0 tcp 150.150.10.2
!
interface loopback 0
ip address 150.150.10.1 255.255.255.0
interface serial 0
 ip address 150.150.11.1 255.255.255.192
!
interface serial 1
 description PU2 with SDLC station role set to secondary
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 9600
 sdlc role primary
 sdlc vmac 4000.9999.0100
 sdlc address 01
 sdlc xid 01 05d20006
 sdlc partner 4000.1020.1000 01
 sdlc dlsw 1
!
interface serial 2
 description Node Type 2.1 with SDLC station role set to negotiable or primary
 encapsulation sdlc
 sdlc role prim-xid-poll
 sdlc vmac 1234.3174.0000
 sdlc address d2
 sdlc partner 1000.5aed.1f53 d2
 sdlc dlsw d2


!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 100 1 2000
 source-bridge spanning
!
interface tokenring 1
 no ip address
 ring-speed 16
 source-bridge 400 1 2000
 source-bridge spanning
!
```

```
router eigrp 202
 network 150.150.0.0
```

DLSw+ over Frame Relay Configuration Example

Frame Relay support extends the DLSw+ capabilities to include Frame Relay in direct mode. Frame Relay support includes permanent virtual circuit capability. DLSw+ runs over Frame Relay with or without local acknowledgement. It supports the Token Ring-to-Token Ring connections similar to FST and other direct data link controls. Figure 138 illustrates a DLSw+ configuration over Frame Relay with RIF Passthrough.

Figure 138 DLSw+ over Frame Relay



The following configuration examples are based on Figure 139. The Token Rings in the illustration are in Ring 2.

Router A

```
source-bridge ring-group 100
dlsw local-peer 10.2.23.1
dlsw remote-peer 0 frame-relay interface serial 0 30 passthru
interface loopback 0
ip address 10.2.23.1 255.255.255.0

interface tokenring 0
 ring-speed 16
 source-bridge spanning 1 1 100
!
interface serial 0
 mtu 3000
 no ip address
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map dlsw 30
```

Router B

```
source-bridge ring-group 100
dlsw local-peer 10.2.23.2
dlsw remote-peer 0 frame-relay interface serial 0 30 passthru
interface loopback 0
ip address 10.2.23.2 255.255.255.0

interface tokenring 0
 ring-speed 16
 source-bridge spanning 2 1 100
!
interface serial 0
 mtu 3000
```

```
no ip address
encapsulation frame-relay
frame-relay lmi-type ansi
frame-relay map dlsw 30
```

DLSw+ over QLLC Configuration Examples

The following three examples describe QLLC support for DLSw+.

Example 1

In this configuration, DLSw+ is used to allow remote devices to connect to a DLSw+ network over an X.25 public packet-switched network.

In this example, all QLLC traffic is addressed to destination address 4000.1161.1234, which is the MAC address of the FEP.

The remote X.25-attached IBM 3174 cluster controller is given a virtual MAC address of 1000.0000.0001. This virtual MAC address is mapped to the X.121 address of the 3174 (31104150101) in the X.25 attached router.

```
interface serial 0
 encapsulation x25
 x25 address 3110212011
 x25 map qllc 1000.0000.0001 31104150101
 qllc dlsw partner 4000.1611.1234
```

Example 2

In this configuration, a single IBM 3174 cluster controller needs to communicate with both an AS/400 and a FEP. The FEP is associated with subaddress 150101 and the AS/400 is associated with subaddress 151102.

If an X.25 call comes in for 33204150101, the call is mapped to the FEP and forwarded to MAC address 4000.1161.1234. The IBM 3174 appears to the FEP as a Token Ring-attached resource with MAC address 1000.0000.0001. The IBM 3174 uses a source SAP of 04 when communicating with the FEP, and a source SAP of 08 when communicating with the AS/400.

```
interface serial 0
 encapsulation x25
 x25 address 31102
 x25 map qllc 1000.0000.0001 33204
 qllc dlsw subaddress 150101 partner 4000.1161.1234
 qllc dlsw subaddress 150102 partner 4000.2034.5678 sap 04 08
```

Example 3

In this example, two different X.25 resources want to communicate over X.25 to the same FEP.

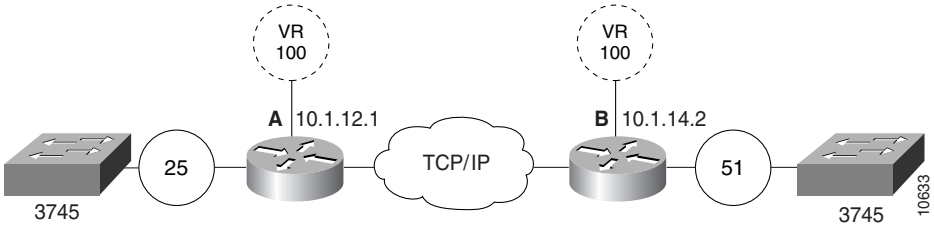
In the router attached to the X.25 network, every X.25 connection request for X.121 address 31102150101 is directed to DLSw+. The first SVC to be established will be mapped to virtual MAC address 1000.0000.0001. The second SVC to be established will be mapped to virtual MAC address 1000.0000.0002.

```
interface serial 0
 encapsulation x25
 x25 address 31102
 x25 map qllc 33204
 x25 map qllc 35765
 qllc dlsw subaddress 150101 vmacaddr 1000.0000.0001 2 partner 4000.1611.1234
```

DLSw+ with RIF Passthrough Configuration Example

Figure 139 is a sample configuration for DLSw+ using the RIF Passthrough feature.

Figure 139 Network Configuration with RIF Passthrough



Router A

```
source-bridge ring-group 100
dlsw local-peer peer id 10.1.12.1
dlsw remote-peer 0 tcp 10.1.14.2 rif-passthru 100
interface loopback 0
ip address 10.1.12.1 255.255.255.0

interface tokenring 0
  ring-speed 16
  source-bridge 25 1 100
  source-bridge spanning
```

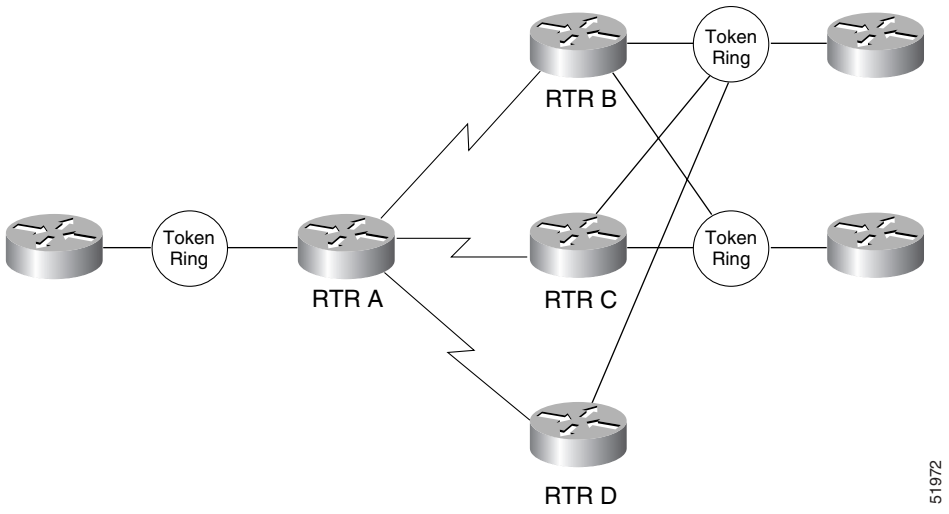
Router B

```
source-bridge ring-group 100
dlsw local-peer peer id 10.1.14.2
dlsw remote-peer 0 tcp 10.1.12.1 rif-passthru 100
interface loopback 0
ip address 10.1.14.2 255.255.255.0

interface tokenring 0
  ring-speed 16
  source-bridge 51 1 100
  source-bridge spanning
```

DLSw+ with Enhanced Load Balancing Configuration Example

Figure 140 shows DLSw+ with the Enhanced Load Balancing feature.

Figure 140 DLSw+ with Enhanced Load Balancing



Router A is configured for the DLSw+ Enhanced Load Balancing feature to load balance traffic among the DLSw+ remote peers B, C, and D.

Router A

```
dlsw local-peer 10.2.19.1
dlsw remote-peer 0 tcp 10.2 24.2 circuit-weight 10
dlsw remote-peer 0 tcp 10.2.19.5 circuit-weight 6
dlsw remote-peer 0 tcp 10.2.20.1 circuit-weight 20
dlsw load-balance circuit-count
dlsw timer explorer-wait-time 100
```

Router B

```
dlsw local-peer 10.2.24.2 cost 1 promiscuous
```

Router C

```
dlsw local-peer 10.2.19.5 cost 1 promiscuous
```

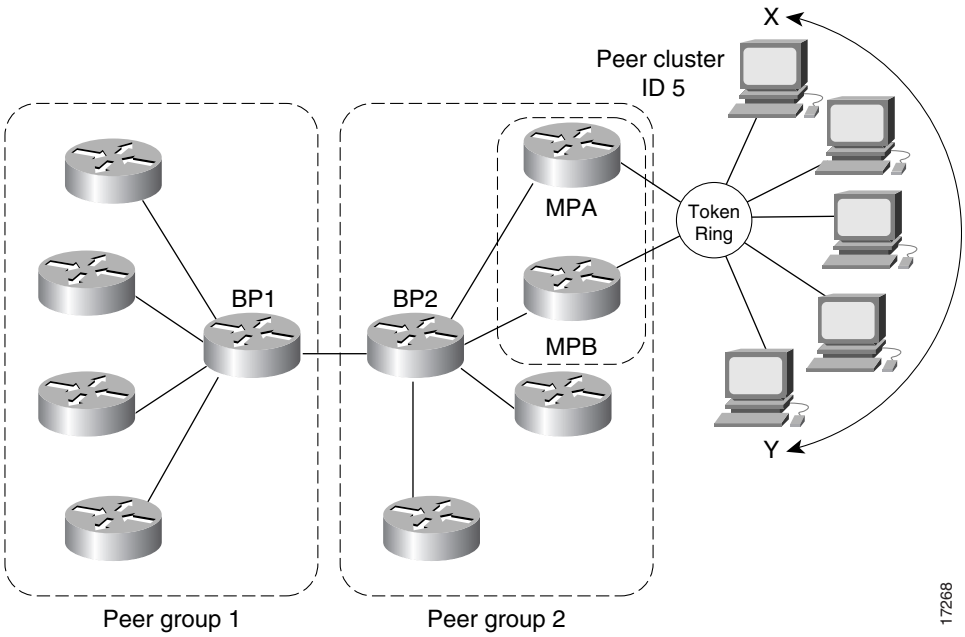
Router D

```
dlsw local-peer 10.2.20.1 cost 1 promiscuous
```

DLSw+ Peer Cluster Feature Configuration Example

Figure 141 shows a DLSw+ network configured with the DLSw+ Peer Clusters feature.

Figure 141 DLSw+ Peer Cluster Feature



Because BP2 is configured as the border peer with the DLSw+ Peer Clusters feature, it does not forward explorers to both MPA and MPB since they are part of the same peer cluster.

BP2

```
source-bridge ring-group 310
dlsw local-peer 10.1.1.3 border group 2 promiscuous
```

MPA

```
source-bridge ring-group 310
dlsw local-peer 10.1.1.1 group 2 promiscuous cluster 5
dlsw remote-peer 0 tcp 10.1.1.3
```

MPB

```
source-bridge ring-group 310
dlsw local-peer 10.1.1.2 group 2 promiscuous cluster 5
dlsw remote-peer tcp 0 10.1.1.3
```

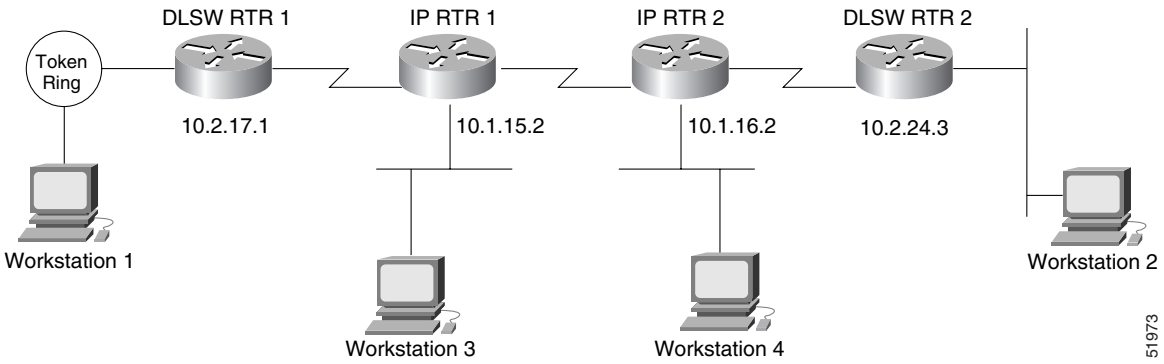
MPC

```
dlsw local-peer 10.1.1.4 group 2 promiscuous
dlsw remote-peer tcp 0 10.1.1.3
```

DLSW+ RSVP Bandwidth Reservation Feature Configuration Example

Figure 142 shows a DLSw+ network with the DLSw+ RSVP Bandwidth Reservation feature configured.

Figure 142 DLSw+ RSVP Bandwidth Reservation Feature Configured



DLSWRTR 1 and DLSWRTR 2 are configured for the DLSw+ RSVP Bandwidth Reservation feature with an average bit rate of 40 and a maximum-burst rate of 10.

DLSWRTR 1

```
dlsw local-peer peer id 10.2.17.1
dlsw remote-peer 0 tcp 10.2.24.3
dlsw rsvp 40 10
```

DLSWRTR2

```
dlsw local-peer peer id 10.2.24.3
dlsw remote-peer 0 tcp 10.2.17.1
dlsw rsvp 40 10
```

The following output of the **show ip rsvp sender** command on the DLSWRTR2 verifies that PATH messages are being sent from DLSWRTR2:

```
DLSWRTR2#show ip rsvp sender
To         From       Pro DPort Sport Prev Hop I/F  BPS     Bytes
10.2.17.1 10.2.24.3 TCP 2065  11003                 10K     28K
10.2.24.3 10.2.17.1 TCP 11003 2065 10.2.17.1 Et1/1 10K     28K
```

The following output of the **show ip rsvp req** command on the DLSWRTR2 verifies that RESV messages are being sent from DLSWRTR2:

```
DLSWRTR2#show ip rsvp req
To             From           Pro DPort Sport Next Hop      I/F    Fi Serv BPS Bytes
10.2.24.3   10.2.17.1       TCP 11003 2065 10.2.17.1     Et1/1 FF RATE 10K   28K
```

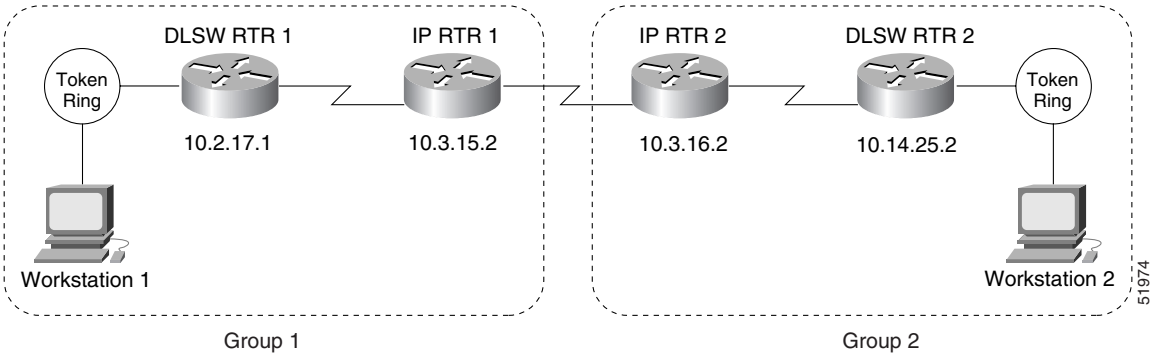
If the IP cloud is able to guarantee the bandwidth requested and the **show ip rsvp sender** and **show ip rsvp req** commands are successful, issue the **show ip rsvp res** command to verify that a reservation was made from DLSWRTR1 to DLSWRTR2:

```
DLSWRTR2#show ip rsvp rese
To             From      Pro DPort Sport Next Hop      I/F    Fi Serv BPS Bytes
10.2.17.1    10.2.24.3 TCP 2065  11003 10.2.17.1 Et1/1 FF RATE     10K   28K
10.2.24.3    10.2.17.1 TCP 11003 2065                  FF RATE     10K   28K
```

DLSw+ RSVP Bandwidth Reservation Feature with Border Peers Configuration Example

Figure 143 shows a DLSw+ border peer network configured with DLSw+ RSVP.

Figure 143 DLSw+ RSVP Bandwidth Reservation Feature in a Border Peer Network



The following example configures DLSWRTR1 to send PATH messages at rates of 40 kbps and 10 kbps and DLSWRTR2 to send PATH messages at rates of 10.

DLSWRTR1

```
dlsw local-peer peer-id 10.2.17.1 group 1 promiscuous
dlsw rsvp default
dlsw remote-peer 0 tcp 10.3.15.2
dlsw peer-on-demand-defaults rsvp 40 10
```

IPRTR1

```
dlsw local-peer peer-id 10.3.15.2 group 1 border promiscuous
dlsw remote-peer 0 tcp 10.3.16.2
```

IPRTR2

```
dlsw local-peer peer-id 10.3.16.2 group 2 border promiscuous
dlsw remote-peer 0 tcp 10.3.15.2
```

DLSWRTR2

```
dlsw local-peer peer-id 10.14.25.2 group 2 promiscuous
dlsw rsvp default
dlsw remote-peer 0 tcp 10.3.16.2
```

The following output of the **show ip rsvp sender** command on DLSWRTR2 verifies that PATH messages are being sent from DLSWRTR2:

```
DLSWRT2#show ip rsvp sender
To              From            Pro DPort Sport Prev Hop        I/F  BPS  Bytes
10.2.17.1       10.14.25.2      TCP 2065  11003                       10K    28K
10.14.25.2      10.2.17.1       TCP 11003 2065 10.2.17.1        Et1/1 10K    28K
```

The following output of the **show ip rsvp request** command on DLSWRTR2 verifies that RESV messages are being sent from DLSWRTR 2:

```
DLSWRT2#show ip rsvp req
To            From          Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
10.14.25.2    10.2.17.1     TCP 11003 2065 10.2.17.1      Et1/1 FF RATE 10K   28K
```

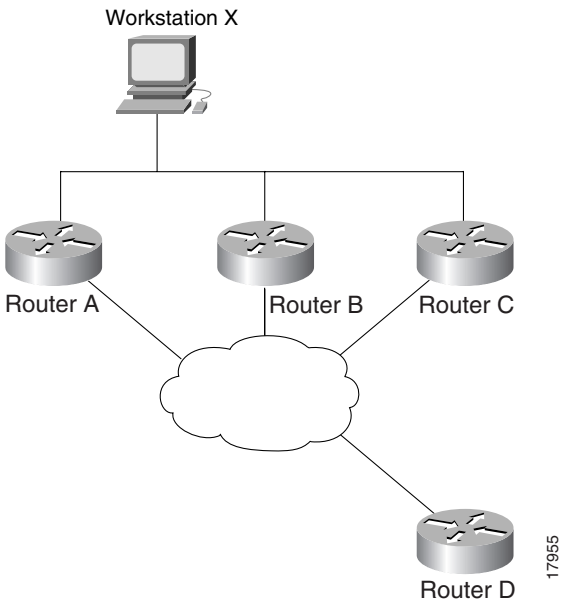
The following output of the **show ip rsvp res** command on the DLSWRTR1 verifies that the RSVP reservation was successful:

```
DLSWRTR1#show ip rsvp rese
To              From            Pro DPort Sport Next Hop      I/F    Fi Serv BPS Bytes
10.2.17.1       10.14.25.2      TCP 2065  11003 10.14.25.2    Et1/1 FF RATE 10K   28K
10.14.25.2      10.2.17.1       TCP 11003 2065                      FF RATE 10K   28K
```

DLSw+ with Ethernet Redundancy Configuration Example

Figure 144 shows that Router A, Router B, and Router C advertise their presence on the Ethernet via their Ethernet interfaces to the multicast MAC address 9999.9999.9999. Because Router B is the master router, it keeps a database of all circuits handled within the domain and grants or denies permission for new circuit requests for Router A and Router C. There is no special configuration required for the end stations or for the remote peer. Only the DLSw+ devices on the LAN need the extra configuration. Master Router B waits 1.5 seconds after it receives the first IWANTIT primitive before assigning the new SNA circuit to one of its ethernet redundancy peers because of the **dlsw transparent timers sna** *1500* command.

Figure 144 DLSw+ with Ethernet Redundancy



Router A

```
dlsw local-peer peer id 10.2.24.2
dlsw remote-peer 0 tcp 10.2.17.1
interface loopback 0
ip address 10.2.24.2 255.255.255.0

int e1
 ip address 150.150.2.1 255.255.255.0
 dlsw transparent redundancy-enable 9999.9999.9999
```

Router B

```
dlsw local-peer peer-id 10.2.24.3
```

```
dlsw remote-peer 0 tcp 10.1.17.1
interface loopback 0
ip address 10.2.24.3 255.255.255.0

int e1
 ip address 150.150.2.2 255.255.255.0
 dlsw transparent redundancy-enable 9999.9999.9999 master priority 1
 dlsw transparent timers sna 1500
```

Router C

```
dlsw local-peer peer-id 10.2.24.4
dlsw remote-peer 0 tcp 10.2.17.1
interface loopback 0
ip address 10.2.24.4 255.255.255.0

int e1
 ip address 150.150.2.3 255.255.255.0
 dlsw transparent redundancy-enable 9999.9999.9999
```

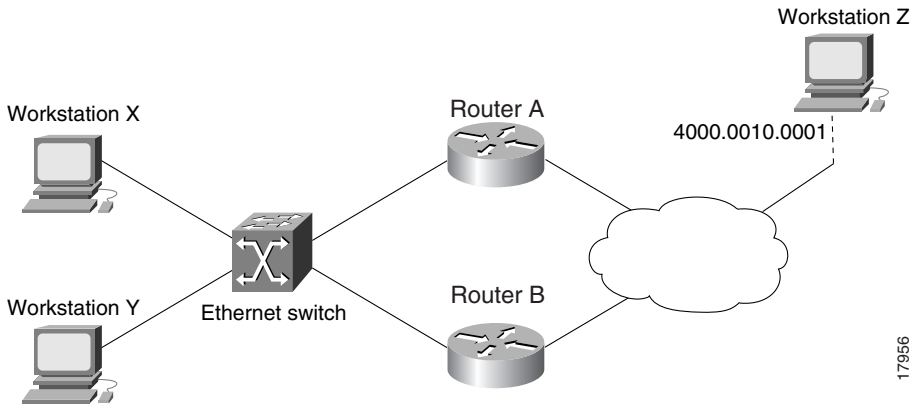
Router D

```
dlsw local-peer peer-id 10.2.17.1 promiscuous
```

DLSw+ with Ethernet Redundancy Enabled for Switch Support Configuration Example

Figure 145 is a sample configuration of the DLSw+ Ethernet Redundancy feature in a switched environment. The ethernet switch sees the device with MAC address 4000.0010.0001 one port at a time because Router A and Router B have mapped different MAC addresses to it. This configuration is known as MAC-address mapping. Router A is configured so that MAC address 4000.0001.0000 maps to the actual device with MAC address 4000.0010.0001. Router B is configured so that MAC address 4000.0201.0001 maps to the actual device with MAC address 4000.0010.0001. Router A and B backup one another. Router A is configured as the master with a default priority of 100. Master Router A waits 1.5 seconds after it receives the first IWANTIT primitive before assigning the new SNA circuit to one of its ethernet redundancy peers because of the **dlsw transparent timers sna** *1500* command.

Figure 145 DLSw+ with Ethernet Redundancy in a Switched Environment



Router A

```
dlsw local peer peer-id 10.2.17.1
dlsw remote-peer 0 tcp 10.3.2.1
dlsw transparent switch-support
interface loopback 0
ip address 10.2.17.1 255.255.255.0

int e 0
 mac-address 4000.0000.0001
 ip address 150.150.2.1 255.255.255.0
 dlsw transparent redundancy-enable 9999.9999.9999 master-priority
 dlsw transparent map local-mac 4000.0001.0000 remote-mac 4000.0010.0001
 neighbor 4000.0000.0011
 dlsw transparent timers sna 1500
```

Router B

```
dlsw local peer peer-id 10.2.17.2
dlsw remote-peer 0 tcp 10.3.2.1
dlsw transport switch-support
interface loopback 0
ip address 10.2.17.2 255.255.255.0

int e 1
 mac-address 4000.0000.0011
 ip address 150.150.3.1 255.255.255.0
 dlsw transparent redundancy-enable 9999.9999.9999
 dlsw transparent map local-mac 4000.0201.0001 remote-mac 4000.0010.0001
 neighbor 4000.0000.0001
```

Configuring Serial Tunnel and Block Serial Tunnel

Bisync-to-IP Conversion for Automated Teller Machines

Feature History

Release	Modification
12.2(4)T	This feature was introduced.
12.2(13)T	The **bstun peer-map-poll** command was added.

This document describes the Bisync-to-IP Conversion for Automated Teller Machines feature in Cisco IOS Release 12.2(4)T and the enhancement introduced in Cisco IOS Release 12.2(13)T and includes the following sections:

Feature Overview

The Bisync-to-IP Conversion for Automated Teller Machines feature enables customers to attach a binary synchronous communication (bisync) automated teller machine to a serial interface on a Cisco router running bisync-to-IP (BIP) protocol translation, and then to route the data over a TCP/IP network directly to an IP-based application host.

This feature works by removing the bisync protocol headers enclosing the application data, creating a TCP/IP connection with the application host, and delivering the data directly to the TCP/IP application running on that host.

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

At the router, application data from the attached bisync device is encapsulated in IP. At the host site, traffic is delivered directly to the application host computer through use of a new command provided with this feature, **bstun route** (BIP). The **bstun route** (BIP) command is used to specify BIP peer tunneling as the method to be used to encapsulate data from a Block Serial Tunnel (BSTUN) interface to a bisync device to a remote host over an IP network. For more information about this command, see the "Command Reference" section of this document.

Figure 1 shows how you can configure the bisync link between two devices so that converted traffic is delivered to a host using BIP.

Figure 1 Cisco Router Support of Bisync Devices with BIP

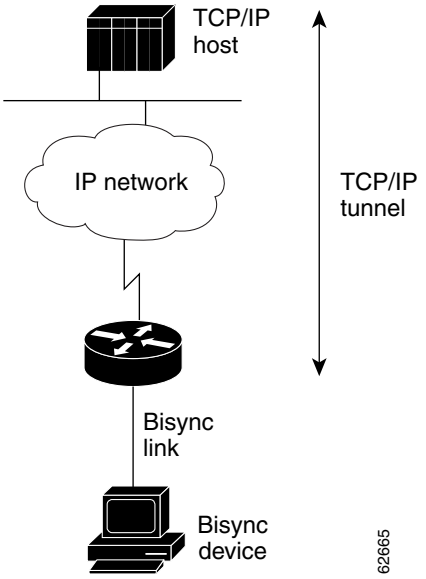
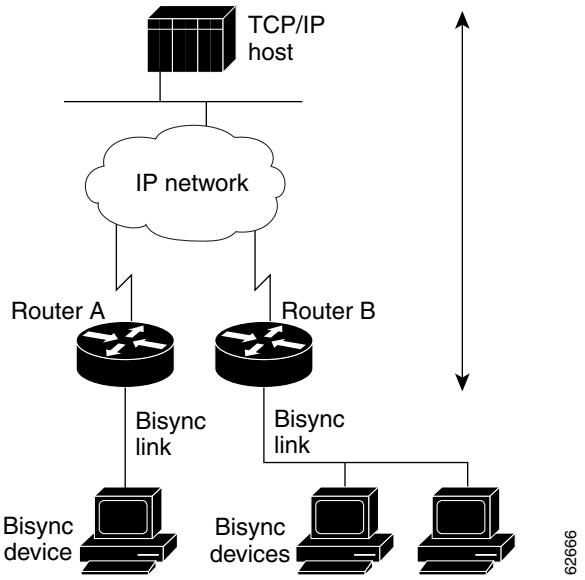


Figure 2 shows how you can configure the bisync link between multiple routers and multiple bisync devices so that converted traffic is delivered to a host using BIP.

Figure 2 Cisco Router Support of Multiple Routers and Multiple Bisync Devices with BIP



This feature is closely associated with BSTUN and the procedures for configuring them. To configure this feature, you will use many of the same procedures and commands required for configuring BSTUN, without the previous requirement of terminating the connection on a peer Cisco router and recreating the bisync serial protocol at the peer. For this feature, rather than one end of the tunnel terminating on another router, the tunnel terminates on a non-Cisco host computer. This feature is designed to address the specific requirements of automated teller machines only, and requires that the host application accommodate this specific method. This restriction is necessary because the Bisync-to-IP Conversion for Automated Teller Machines feature provides conversion for a single device within a 3270 control unit.

For more information, see the "Prerequisites" section and the "Configuration Tasks" section later in this document.

Mapping the Peer State to Polling

When using BIP, Automated Teller Machines (the peer devices) are polled when the BIP tunnel between the host machine and the Automated Teller Machine becomes active. At the same time the Automated Teller Machine is powered on and could be active. Because the state of the host application is unknown at this point, there can be a window in which the host is not ready to receive anything from the Automated Teller Machine even though the Automated Teller Machine is active.

As of Cisco IOS Release 12.2(13)T you can use the **bstun peer-map-poll** command in global configuration mode to map the Automated Teller Machine state to polling. The default is to not map the peer state to polling. If you configure this command, BIP activates polling when the BIP tunnel becomes active and stops polling when the tunnel connection is terminated. When the peer state-to-polling is not mapped, BIP waits for the host to issue an "active" status message across the BIP tunnel before polling the Automated Teller Machine device and polling is stopped when an "inactive" status message is received across the tunnel or the tunnel connection is terminated.

Benefits

Improves System Performance

This feature has the advantage of removing complexity from central data centers, thus reducing cost and enabling a data center to more easily use multiservice applications. At the same time, it improves system performance by removing potential points of failure and allowing for multiple paths for delivery of the application data.

Accommodates Improved Network Design

This feature supports the conversion of bisync to native TCP for direct delivery to host applications enabled for TCP devices. This enhanced support eliminates the need for headend tunnel routers, and allows for network designs that provide higher availability. It allows the remote, serially attached bisync device to attach to the application host through a LAN interface instead of through a serial interface.

Restrictions

This feature addresses the specific requirements of automated teller machines only. It requires that the host application accommodate removing the BIP headers encapsulating the application data. This restriction is necessary because the Bisync-to-IP Conversion for Automated Teller Machines feature provides conversion for a single device within a 3270 control unit.

Related Features and Technologies

- Bridging and IBM networking
- BSTUN

Related Documents

- *Cisco IOS Bridging and IBM Networking Configuration Guide*, Release 12.2
- *Cisco IOS Bridging and IBM Networking Command Reference*, Release 12.2

Supported Platforms

- Cisco 1600 series
- Cisco 1700 series
- Cisco 2500 series
- Cisco 2600 series
- Cisco 3600 series
- Cisco 7200 series

Determining Platform Support Through Cisco Feature Navigator

Cisco IOS software is packaged in feature sets that are supported on specific platforms. To get updated information regarding platform support for this feature, access Cisco Feature Navigator. Cisco Feature Navigator dynamically updates the list of supported platforms as new platform support is added for the feature.

Cisco Feature Navigator is a web-based tool that enables you to determine which Cisco IOS software images support a specific set of features and which features are supported in a specific Cisco IOS image. You can search by feature or release. Under the release section, you can compare releases side by side to display both the features unique to each software release and the features in common.

To access Cisco Feature Navigator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to cco-locksmith@cisco.com. An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions found at this URL:

http://www.cisco.com/register

Cisco Feature Navigator is updated regularly when major Cisco IOS software releases and technology releases occur. For the most current information, go to the Cisco Feature Navigator home page at the following URL:

http://www.cisco.com/go/fn

Availability of Cisco IOS Software Images

Platform support for particular Cisco IOS software releases is dependent on the availability of the software images for those platforms. Software images for some platforms may be deferred, delayed, or changed without prior notice. For updated information about platform support and availability of software images for each Cisco IOS software release, refer to the online release notes or, if supported, Cisco Feature Navigator.

Note To find the releases and platforms associated with this feature, choose BIP from the list of available features.

Supported Standards, MIBs, and RFCs

Standards

No new or modified standards are supported by this feature.

MIBs

- CISCO-BSTUN-MIB

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

No new or modified standards are supported by this feature.

Prerequisites

Before you configure the Bisync-to-IP Conversion for Automated Teller Machines feature, BSTUN must be enabled. For more information about BSTUN, refer to the following chapters of the *Cisco IOS Bridging and IBM Networking Configuration Guide*, Release 12.2:

- "Overview of IBM Networking" chapter
- "Configuring Serial Tunnel and Block Serial Tunnel" chapter (specifically, the "Block Serial Tunneling (BSTUN) Overview" and the "BSTUN Configuration Task List" sections)

Configuration Tasks

See the following sections for configuration tasks for the Bisync-to-IP Conversion for Automated Teller Machines feature. Each task in the list is identified as either required or optional.

- Enabling BSTUN (required)
- Mapping the Peer State to Polling (optional)
- Defining the Protocol Group (required)
- Setting the Reconnect Interval (required)
- Enabling BSTUN Remote Keepalive (required)

- Configuring BSTUN on the Serial Interface (required)
- Assigning a Serial Interface to a BSTUN Group (required)
- Configuring Bisync Options on a Serial Interface (required)
- Specifying How Frames Are Forwarded (required)
- Verifying the Status of BSTUN (optional)

Enabling BSTUN

To enable BSTUN in IP networks, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun peer-name ip-address`	Enables BSTUN.

The IP address in the **bstun peer-name** command defines the address by which this BSTUN peer is known to other BSTUN peers that are using the TCP transport. If this command is unconfigured or the **no** form of this command is specified, all BSTUN routing commands with IP addresses are deleted. BSTUN routing commands without IP addresses are not affected by this command.

Mapping the Peer State to Polling

To map the peer state to polling, us the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun peer-map-poll`	Enables BSTUN to map the peer state or received status messages to polling.

Defining the Protocol Group

To define the protocol group, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun protocol-group group-number bsc-local-ack`	Defines the protocol group.

The **bsc-local-ack** keyword is the only one supported by BIP.

Setting the Reconnect Interval

To set the amount of time for the system to wait before trying to reconnect to a peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun reconnect-interval time-value`	Sets the amount of time the system waits before trying to reconnect to the peer. This command applies only to BSTUN route BIP connections that are defined as active.

Enabling BSTUN Remote Keepalive

To enable detection of the loss of a peer, use the following commands in global configuration mode, as needed:

:

Command	Purpose
`Router(config)# bstun remote-peer-keepalive seconds`	Enables detection of the loss of a peer.
`Router(config)# bstun keepalive-count`	Specifies the number of times to attempt a peer connection.

Configuring BSTUN on the Serial Interface

Configure BSTUN on the serial interface before issuing any further BSTUN or protocol configuration commands for the interface. To configure the BSTUN function on a serial interface, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# interface serial number`	Specifies a serial port.
Step 2	`Router(config-if)# encapsulation bstun`	Configures BSTUN on an interface.

Note Configure the **encapsulation bstun** command on an interface before configuring any other BSTUN commands for the interface.

Assigning a Serial Interface to a BSTUN Group

Each BSTUN-enabled interface on a router must be assigned to a previously defined BSTUN group. Packets will travel only between BSTUN-enabled interfaces that are in the same group. To assign a serial interface to a BSTUN group, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bstun group group-number`	Assigns a serial interface to a BSTUN group.

Configuring Bisync Options on a Serial Interface

To configure bisync options on a serial interface, use the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bsc char-set {ascii | ebcdic}`	Specifies the character set used by the bisync support feature.
`Router(config-if)# full-duplex`	Specifies that the interface can run bisync in full-duplex mode.
`Router(config-if)# bsc pause time`	Specifies the amount of time (in tenths of a second) between the start of one polling cycle and the next. • The default value is 10 (that is, 10 tenths of a second, or 1 second). The maximum time is 255 tenths of a second (25.5 seconds).
`Router(config-if)# bsc poll-timeout time`	Specifies the timeout for a poll or a select sequence.
`Router(config-if)# bsc primary`	Specifies that the router is acting as the primary end of the bisync link.
`Router(config-if)# bsc retries retry-count`	Specifies the number of connection attempts before a device is considered to have failed.
`Router(config-if)# bsc servlim servlim-count`	Specifies the number of cycles of the active poll list that are performed between polls to control units in the inactive poll list.

Specifying How Frames Are Forwarded

To specify how frames are forwarded when received on a BSTUN interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bstun route {address cu-address} {bip ip-address} {fport port-number} {lport port-number | passive} [tcp-queue-max] [transparent]`	Propagates the serial frame that contains a specific address. The BIP form of TCP encapsulation is used to propagate the serial frames.

Verifying the Status of BSTUN

To display statistics for BSTUN interfaces, protocol groups, number of packets sent and received, local acknowledgment states, and other activity information, use the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# show bstun [group bstun-group-number] [address address-list]`	Displays the current status of STUN connections.
`Router# show bsc [group bstun-group-number] [address address-list]`	Displays status of the interfaces on which bisync is configured.
`Router# show interfaces`	Displays the current status information for the interface.
`Router# debug bstun events`	Displays BSTUN connection events and status.
`Router# debug bstun packets`	Displays packet information on packets traveling through the BSTUN links.

Configuration Examples

This section provides the following BSTUN traffic routing example. BSTUN traffic with the control unit address C5 is routed to and from the host computer specified by the IP address 192.168.60.100. The BIP form of TCP encapsulation (as indicated by the **bip** keyword) is used to propagate the serial frames.

```
bstun route address C5 bip 192.168.60.100 fport 2000 lport 3005
```

Command Reference

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Bridging Command Reference* at http://www.cisco.com/en/US/docs/ios/bridging/command/reference/br_book.html or the *Cisco IOS IBM Networking Command Reference* at http://www.cisco.com/en/US/docs/ios/ibm/command/reference/ibm_book.html. For information about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

New Commands

- **bstun peer-map-poll**
- **bstun reconnect-interval**
- **bstun route (BIP)**

Modified Command

- **show bstun**

Configuring Serial Tunnel and Block Serial Tunnel

This chapter describes how to configure serial tunnel (STUN) and block serial tunnel (BSTUN). For a complete description of the STUN and BSTUN commands in this chapter, refer to the "STUN and BSTUN Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference*, Volume 1 of 2. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page lv in the "Using Cisco IOS Software" chapter.

Serial Tunnel Overview

Cisco's STUN implementation allows Synchronous Data Link Control (SDLC) protocol devices and High-Level Data Link Control (HDLC) devices to connect to one another through a multiprotocol internetwork rather than through a direct serial link. STUN encapsulates SDLC frames in either the Transmission Control Protocol/Internet Protocol (TCP/IP) or the HDLC protocol. STUN provides a straight passthrough of all SDLC traffic (including control frames, such as Receiver Ready) end-to-end between Systems Network Architecture (SNA) devices.

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

Cisco's SDLC local acknowledgment provides local termination of the SDLC session so that control frames no longer travel the WAN backbone networks. This means end nodes do not time out, and a loss of sessions does not occur. You can configure your network with STUN, or with STUN and SDLC local acknowledgment. To enable SDLC local acknowledgment, the Cisco IOS software must first be enabled for STUN and routers configured to appear on the network as primary or secondary SDLC nodes. TCP/IP encapsulation must be enabled. Cisco's SDLC Transport feature also provides priority queueing for TCP encapsulated frames.

Cisco's BSTUN implementation enables Cisco series 2500, 4000, 4500, 4700 and 7200 series routers to support devices that use the Binary Synchronous Communications (Bisync) data-link protocol and asynchronous security protocols that include Adplex, ADT Security Systems, Inc., Diebold, and asynchronous generic traffic. BSTUN implementation is also supported on the 4T network interface module (NIM) on the Cisco router 4000 and 4500 series. Our support of the Bisync protocol enables enterprises to transport Bisync traffic and SNA multiprotocol traffic over the same network.

STUN Configuration Task List

To configure and monitor STUN or STUN local acknowledgment, perform the tasks in the following sections:

The "STUN Configuration Examples" section on page 15 follows these configuration tasks.

Enabling STUN

To enable STUN, use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun peer-name ip-address`	Enables STUN for a particular IP address.

When configuring redundant links, ensure that the STUN peer names you choose on each router are the IP addresses of the most stable interfaces on each device, such as a loopback or Ethernet interface. See the "STUN Configuration Examples" section on page 15.

You must also configure SDLC address FF on Router A for each of the STUN peers. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# stun route address address-number tcp ip-address [local-ack] [priority] [tcp-queue-max] [passive]`	Configures SDLC address FF on Router A for each STUN peer.

Specifying STUN Protocol Group

Place each STUN interface in a group that defines the ISO 3309-compliant framed protocol running on that link. Packets will only travel between STUN interfaces that are in the same protocol group.

There are three predefined STUN protocols:

- Basic
- SDLC
- SDLC transmission group (TG)

You can also specify a custom STUN protocol.

To specify STUN protocols, you must perform the tasks in the following sections:

- Specifying a Basic STUN Group, page 3
- Specifying an SDLC Group, page 4
- Specifying an SDLC Transmission Group, page 4
- Creating and Specifying a Custom STUN Protocol, page 4

If you want to use the STUN Local Acknowledgment feature, you must specify either the SDLC protocol or the SDLC TG protocol.

Note Before you can specify a custom protocol, you must first define the protocol; see the "Creating and Specifying a Custom STUN Protocol" section on page 4 for the procedure.

Specifying a Basic STUN Group

The basic STUN protocol does not depend on the details of serial protocol addressing and is used when addressing is not important. Use this when your goal is to replace one or more sets of point-to-point (not multidrop) serial links by using a protocol other than SDLC. Use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun protocol-group group-number basic`	Specifies a basic protocol group and assigns a group number.

Specifying an SDLC Group

You can specify SDLC protocol groups to associate interfaces with the SDLC protocol. Use the SDLC STUN protocol to place the routers in the midst of either point-to-point or multipoint (multidrop) SDLC links. To define an SDLC protocol group, enter the following command in global configuration mode:

Command	Purpose
`Router(config)# stun protocol-group group-number sdlc`	Specifies an SDLC protocol group and assigns a group number.

If you specify an SDLC protocol group, you cannot specify the **stun route all** command on any interface of that group.

For an example of how to configure an SDLC protocol group, see the "Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example" section on page 17.

Specifying an SDLC Transmission Group

An SNA TG is a set of lines providing parallel links to the same pair of SNA front-end-processor (FEP) devices. This provides redundancy of paths for fault tolerance and load sharing. To define an SDLC TG, use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun protocol-group group-number sdlc sdlc-tg`	Specifies an SDLC protocol group, assigns a group number, and creates an SNA transmission group.

All STUN connections in a TG must connect to the same IP address and use the SDLC local acknowledgment feature.

Creating and Specifying a Custom STUN Protocol

To define a custom protocol and tie STUN groups to the new protocol, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# stun schema name offset constant-offset length address-length format format-keyword`	Creates a custom protocol.
Step 2	`Router(config)# stun protocol-group group-number schema`	Specifies the custom protocol group and assigns a group number.

Enabling STUN Keepalive

To define the number of times to attempt a peer connection before declaring the peer connection to be down, use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun keepalive-count`	Specifies the number of times to attempt a peer connection.

Enabling STUN Remote Keepalive

To enable detection of the loss of a peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun remote-peer-keepalive` *seconds*	Enables detection of the loss of a peer.

Enabling STUN Quick-Response

You can enable STUN quick-response, which improves network performance when used with local acknowledgment. When STUN quick-response is used with local acknowledgment, the router responds to an exchange identification (XID) or a Set Normal Response Mode (SNRM) request with a Disconnect Mode (DM) response when the device is not in the CONNECT state. The request is then passed to the remote router and, if the device responds, the reply is cached. The next time the device is sent an XID or SNRM, the router replies with the cached DM response.

Note Using STUN quick-response avoids an AS/400 line reset problem by eliminating the Non-Productive Receive Timer (NPR) expiration in the AS/400. With STUN quick-response enabled, the AS/400 receives a response from the polled device, even when the device is down. If the device does not respond to the forwarded request, the router continues to respond with the cached DM response.

To enable STUN quick-response, use the following command in global configuration mode:

Command	Purpose
`Router(config)# stun quick-response`	Enables STUN quick-response.

Enabling STUN Interfaces

Caution When STUN encapsulation is enabled or disabled on an RSP platform, the memory reallocates memory pools (recarve) and the interface shuts down and restarts. The recarve is caused by the change from STUN to another protocol, which results in a change in the MTU size. No user configuration is required.

You must enable STUN on serial interfaces and place these interfaces in the protocol groups you have defined. To enable STUN on an interface and to place the interface in a STUN group, use the following commands in interface configuration mode:

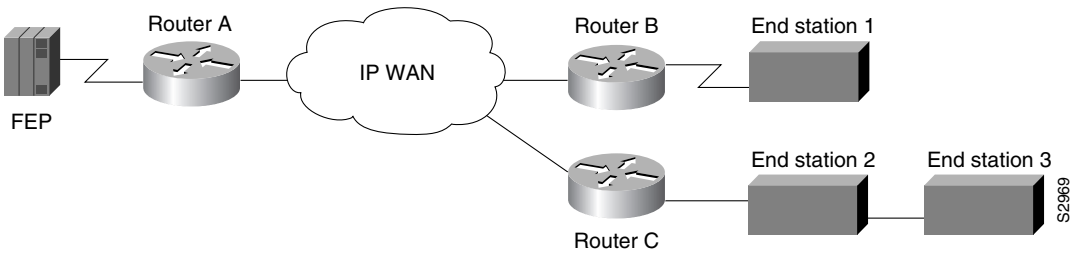
	Command	Purpose
Step 1	`Router(config-if)# encapsulation stun`	Enables STUN function on a serial interface.
Step 2	`Router(config-if)# stun group` *group-number*	Places the interface in a previously defined STUN group.

When a given serial link is configured for the STUN function, it is no longer a shared multiprotocol link. All traffic that arrives on the link will be transported to the corresponding peer as determined by the current STUN configuration.

Configuring SDLC Broadcast

The SDLC broadcast feature allows SDLC broadcast address FF to be replicated for each of the STUN peers, so each of the end stations receives the broadcast frame. For example, in Figure 146, the FEP views the end stations 1, 2, and 3 as if they are on an SDLC multidrop link. Any broadcast frame sent from the FEP to Router A is duplicated and sent to each of the downstream routers (B and C).

Figure 146 SDLC Broadcast across Virtual Multidrop Lines



To enable SDLC broadcast, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc virtual-multidrop`	Enables SDLC broadcast.

Only enable SDLC broadcast on the device that is configured to be the secondary station on the SDLC link (Router A in Figure 146).

Establishing the Frame Encapsulation Method

To allow SDLC frames to travel across a multimedia, multiprotocol network, you must encapsulate them using one of the methods in the following sections:

Configuring HDLC Encapsulation without Local Acknowledgment

You can encapsulate SDLC or HDLC frames using the HDLC protocol. The outgoing serial link can still be used for other kinds of traffic. The frame is not TCP encapsulated. To configure HDLC encapsulation, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# stun route all interface serial number`	Forwards all HDLC or SDLC traffic of the identified interface number.
or	or
`Router(config-if)# stun route all interface serial number direct`	Forwards all HDLC or SDLC traffic on a direct STUN link.
or	or
`Router(config-if)# stun route address address-number interface serial number`	Forwards HDLC or SDLC traffic of the identified address.
or	or
`Router(config-if)# stun route address address-number interface serial number direct`	Forwards HDLC or SDLC traffic of the identified address across a direct STUN link.

Use the **no** forms of these commands to disable HDLC encapsulation.

Note You can forward all traffic only when you are using basic STUN protocol groups.

Configuring TCP Encapsulation without Local Acknowledgment

If you do not want to use SDLC local acknowledgment and only need to forward all SDLC frames encapsulated in TCP, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# stun route all tcp ip-address [passive]`	Forwards all TCP traffic for this IP address.
Step 2	`Router(config-if)# stun route address address-number tcp ip-address [local-ack] [priority] [tcp-queue-max] [passive]`	Specifies TCP encapsulation.

Use the **no** form of these commands to disable forwarding of all TCP traffic.

This configuration is typically used when two routers can be connected via an IP network as opposed to a point-to-point link.

Configuring TCP Encapsulation with SDLC Local Acknowledgment and Priority Queueing

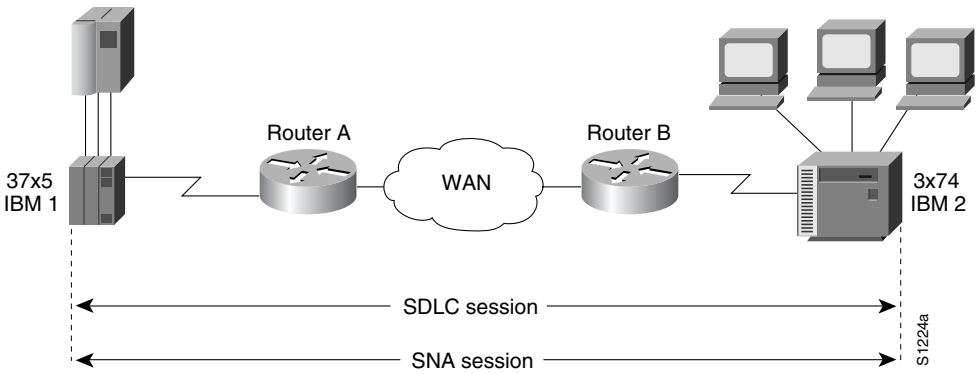
You configure SDLC local acknowledgment using TCP encapsulation. When you configure SDLC local acknowledgment, you also have the option to enable support for priority queueing.

Note To enable SDLC local acknowledgment, you must specify an SDLC or SDLC TG.

SDLC local acknowledgment provides local termination of the SDLC session so that control frames no longer travel the WAN backbone networks. This means that time-outs are less likely to occur.

Figure 147 illustrates an SDLC session. IBM 1, using a serial link, can communicate with IBM 2 on a different serial link separated by a wide-area backbone network. Frames are transported between Router A and Router B using STUN, but the SDLC session between IBM 1 and IBM 2 is still end-to-end. Every frame generated by IBM 1 traverses the backbone network to IBM 2, which, upon receipt of the frame, acknowledges it.

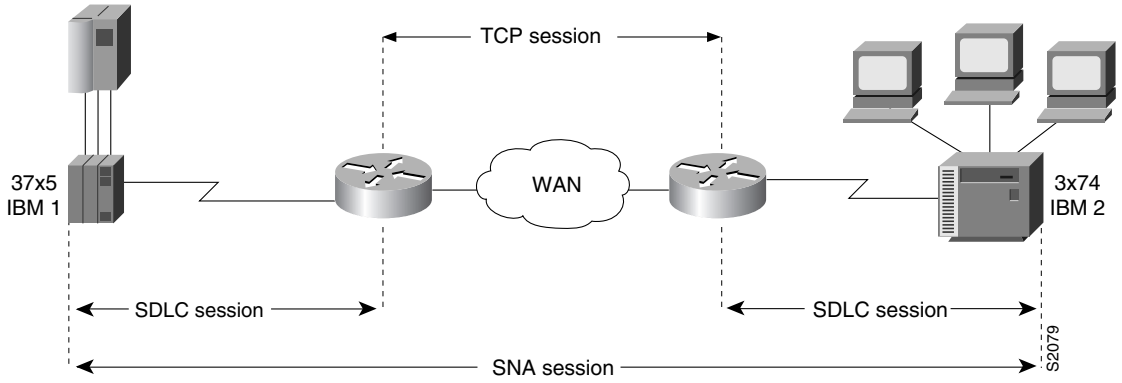
Figure 147 SDLC Session Without Local Acknowledgment



With SDLC local acknowledgment, the SDLC session between the two end nodes is not end-to-end, but instead terminates at the two local routers, as shown in Figure 148. The SDLC session with IBM 1 ends at Router A, and the SDLC session with IBM 2 ends at Router B. Both Router A and Router B execute the full SDLC protocol as part of SDLC Local Acknowledgment. Router A acknowledges frames

received from IBM 1. The node IBM 1 treats the acknowledgments it receives as if they are from IBM 2. Similarly, Router B acknowledges frames received from IBM 2. The node IBM 2 treats the acknowledgments it receives as if they are from IBM 1.

Figure 148 SDLC Session with Local Acknowledgment



To configure TCP encapsulation with SDLC local acknowledgment and priority queueing, perform the tasks in the following sections:

- Assigning the Router an SDLC Primary or Secondary Role, page 9
- Enabling the SDLC Local Acknowledgment Feature, page 10
- Establishing Priority Queueing Levels, page 10

Assigning the Router an SDLC Primary or Secondary Role

To establish local acknowledgment, the router must play the role of an SDLC primary or secondary node. Primary nodes poll secondary nodes in a predetermined order. Secondaries then send if they have outgoing data.

For example, in an IBM environment, an FEP is the primary station and cluster controllers are secondary stations. If the router is connected to an FEP, the router should appear as a cluster controller and must be assigned the role of a secondary SDLC node. If the router is connected to a cluster controller, the router should appear as an FEP and must be assigned the role of a primary SDLC node. Devices connected to SDLC primary end-stations must play the role of an SDLC secondary and routers attached to SDLC secondary end stations must play the role of an SDLC primary station.

To assign the router a primary or secondary role, use one of the following commands in interface configuration mode, as needed:

Command	Purpose

`Router(config-if)# stun sdlc-role primary`
 	
or
 	
`Router(config-if)# stun sdlc-role secondary`
	Assigns the STUN-enabled router an SDLC primary role.
	or
	Assigns the STUN-enabled router an SDLC secondary role.

Enabling the SDLC Local Acknowledgment Feature

To enable SDLC local acknowledgment, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# stun route address address-number tcp ip-address [local-ack] [priority] [tcp-queue-max] [passive]`	Establishes SDLC local acknowledgment using TCP encapsulation.

The **stun route address 1 tcp local-ack priority tcp-queue-max** interface configuration command enables local acknowledgment and TCP encapsulation. Both options are required to use TGs. You should specify the SDLC address with the echo bit turned off for TG interfaces. The SDLC broadcast address 0xFF is routed automatically for TG interfaces. The **priority** keyword creates multiple TCP sessions for this route. The **tcp-queue-max** keyword sets the maximum size of the outbound TCP queue for the SDLC. The default TCP queue size is 100. The value for **hold-queue in** should be greater than the value for **tcp-queue-max**.

You can use the **priority** keyword (to set up the four levels of priorities to be used for TCP encapsulated frames) at the same time you enable local acknowledgment. The **priority** keyword is described in the following section. Use the **no** form of this command to disable SDLC Local Acknowledgment. For an example of how to enable local acknowledgment, see the "Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example" section on page 17.

Establishing Priority Queueing Levels

With SDLC local acknowledgment enabled, you can establish priority levels used in priority queueing for serial interfaces. The priority levels are as follows:

- Low
- Medium
- Normal
- High

To set the priority queueing level, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# stun route address address-number tcp ip-address [local-ack] priority [tcp-queue-max] [passive]`	Establishes the four levels of priorities to be used in priority queueing.

Use the **no** form of this command to disable priority settings. For an example of how to establish priority queueing levels, see the "Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example" section on page 17.

Configuring Local Acknowledgment for Direct Frame Relay Connectivity

To implement STUN with local acknowledgment using direct Frame Relay encapsulation, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# stun route address` *sdlc-addr* `interface` *frame-relay-port* `dlci` number *localsap* `local-ack cls`	Configures Frame Relay encapsulation between STUN peers with local acknowledgment.

Configuring STUN with Multilink Transmission Groups

You can configure multilink SDLC TGs across STUN connections between IBM communications controllers such as IBM 37x5s. Multilink TGs allow you to collapse multiple WAN leased lines into one leased line.

SDLC multilink TGs provide the following features:

- Network Control Program (NCP) SDLC address allowances, including echo and broadcast addressing.
- Remote NCP load sequence. After a SIM/RIM exchange but before a SNRM/UA exchange, NCPs send numbered I-frames. During this period, I-frames are not locally acknowledged, but instead are passed through. After the SNRM/UA exchange, local acknowledgment occurs.
- Rerouting of I-frames sent by the Cisco IOS software to the NCP if a link is lost in a multilink TG.
- Flow control rate tuning causes a sending NCP to “feel” WAN congestion and hold frames that would otherwise be held by the Cisco IOS software waiting to be sent on the WAN. This allows the NCP to perform its class-of-service algorithm more efficiently based on a greater knowledge of network congestion.

STUN connections that are part of a TG must have local acknowledgment enabled. Local acknowledgment keeps SDLC poll traffic off the WAN and reduces store-and-forward delays through the router. It also might minimize the number of NCP timers that expire due to network delay. Also, these STUN connections must go to the same IP address. This is because SNA TGs are parallel links between the same pair of IBM communications controllers.

Design Recommendations

This section provides some recommendations that are useful in configuring SDLC multilink TGs.

The bandwidth of the WAN should be larger than or equal to the aggregate bandwidth of all serial lines to avoid excessive flow control and to ensure response timed does not degrade. If other protocols are also using the WAN, ensure that the WAN bandwidth is significantly greater than the aggregate SNA serial line bandwidth to ensure that the SNA traffic does not monopolize the WAN.

When you use a combination of routed TGs and directly connected NCP TGs, you need to plan the configuration carefully to ensure that SNA sessions do not stop unexpectedly. Assuming that hardware reliability is not an issue, single-link routed TGs are as reliable as direct NCP-to-NCP single-link TGs. This is true because neither the NCP nor the Cisco IOS software can reroute I-frames when a TG has only one link. Additionally, a multilink TG directed between NCPs and a multilink TG through a router are equally reliable. Both can perform rerouting.

However, you might run into problems if you have a configuration in which two NCPs are directly connected (via one or more TG links) and one link in the TG is routed. The NCPs treat this as a multilink TG. However, the Cisco IOS software views the TG as a single-link TG.

A problem can arise in the following situation: Assume that an I-frame is being sent from NCP A (connected to router A) to NCP B (connected to router B) and that all SDLC links are currently active. Router A acknowledges the I-frame sent from NCP A and sends it over the WAN. If, before the I-frame reaches Router B, the SDLC link between router B and NCP B goes down, Router B attempts to reroute the I-frame on another link in the TG when it receives the I-frame. However, because this is a single-link TG, there are no other routes, and Router B drops the I-frame. NCP B never receives this I-frame because Router A acknowledges its receipt, and NCP A marks it as sent and deletes it. NCP B detects a gap in the TG sequence numbers and waits to receive the missing I-frame. NCP B waits forever for this I-frame, and does not send or receive any other frames. NCP B is technically not operational and all SNA sessions through NCP B are lost.

Finally, consider a configuration in which one or more lines of an NCP TG are connected to a router and one or more lines are directly connected between NCPs. If the network delay associated with one line of an NCP TG is different from the delay of another line in the same NCP TG, the receiving NCP spends additional time resequencing PIUs.

Setting Up STUN Traffic Priorities

To determine the order in which traffic should be handled on the network, use the methods described in the following sections:

- Assigning Queueing Priorities, page 12
- Prioritizing STUN Traffic over All Other Traffic, page 14

Assigning Queueing Priorities

To assign queueing priorities, perform the tasks in one of the following sections:

- Prioritizing by Serial Interface Address or TCP Port, page 12
- Prioritizing by Logical Unit Address, page 13

Prioritizing by Serial Interface Address or TCP Port

You can prioritize traffic on a per-serial-interface address or TCP port basis. You might want to do this so that traffic between one source-destination pair is always sent before traffic between another source-destination pair.

Note You must first enable local acknowledgment and priority levels as described earlier in this chapter.

To prioritize traffic, use one of the following commands in global configuration mode, as needed:

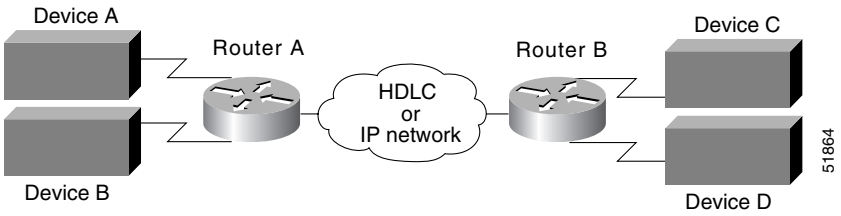
Command	Purpose
`Router(config)# priority-list list-number protocol stun queue address group-number address-number`	Assigns a queueing priority to the address of the STUN serial interface.
or	or
`Router(config)# priority-list list-number protocol ip queue tcp tcp-port-number`	Assigns a queueing priority to a TCP port.

You must also use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# priority-group list-number`	Assigns a priority list to a priority group.

Figure 149 illustrates serial link address prioritization. Device A communicates with Device C, and Device B communicates with Device D. With the serial link address prioritization, you can choose to give A-C a higher priority over B-D across the serial tunnel.

Figure 149 Serial Link Address Prioritization



To disable priorities, use the **no** forms of these commands.

For an example of how to prioritize traffic according to serial link address, see the "Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example" section on page 17.

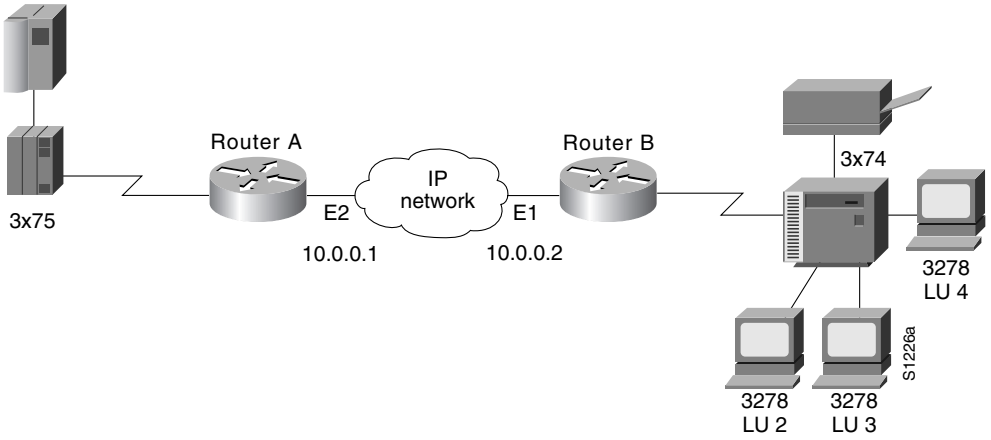
Prioritizing by Logical Unit Address

SNA local logical unit (LU) address prioritization is specific to IBM SNA connectivity and is used to prioritize SNA traffic on either STUN or remote source-route bridging (RSRB). To set the queueing priority by LU address, use the following command in global configuration mode:

Command	Purpose
`Router(config)# locaddr-priority-list list-number address-number queue-keyword`	Assigns a queueing priority based on the LU address.

In Figure 150, LU address prioritization can be set so that particular LUs receive data in preference to others or so that LUs have priority over the printer, for example.

Figure 150 SNA LU Address Prioritization



To disable this priority, use the **no** form of this command.

For an example of how to prioritize traffic according to logical unit address, see the "LOCADDR Priority Groups for STUN Example" section on page 23.

Prioritizing STUN Traffic over All Other Traffic

You can prioritize STUN traffic to be routed first before all other traffic on the network. To give STUN traffic this priority, use the following command in global configuration mode:

Command	Purpose
`Router(config)# priority-list` *list-number* `protocol stun` *queue* `address` *group-number address-number*	Prioritizes STUN traffic in your network over that of other protocols.

To disable this priority, use the **no** form of this command.

For an example of how to prioritize STUN traffic over all other traffic, see the "Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example" section on page 17.

Monitoring and Maintaining STUN Network Activity

You can list statistics regarding STUN interfaces, protocol groups, number of packets sent and received, local acknowledgment states, and more. To get activity information, use the following command in privileged EXEC mode:

Command	Purpose
`Router# show stun`	Lists the status display fields for STUN interfaces.

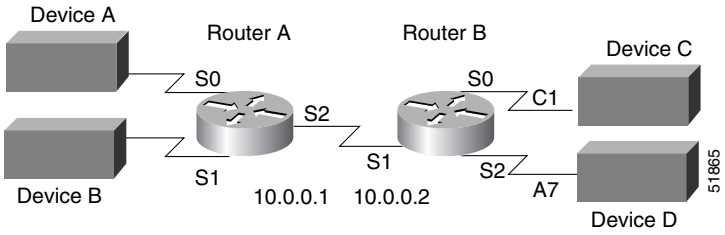
STUN Configuration Examples

The following sections provide STUN configuration examples:

STUN Priorities Using HDLC Encapsulation Example

Assume that the link between Router A and Router B in Figure 151 is a serial tunnel that uses the simple serial transport mechanism. Device A communicates with Device C (SDLC address C1) with a high priority. Device B communicates with Device D (SDLC address A7) with a normal priority.

Figure 151 STUN Simple Serial Transport



The following configurations set the priority of STUN hosts A, B, C, and D.

Router A

```
stun peer-name 10.0.0.1
stun protocol-group 1 sdlc
stun protocol-group 2 sdlc
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 interface serial 2
!
interface serial 1
 no ip address
 encapsulation stun
 stun group 2
 stun route address A7 interface serial 2
!


interface serial 2
 ip address 10.0.0.1 255.0.0.0
```

```
 priority-group 1
!
priority-list 1 protocol stun high address 1 C1
priority-list 1 protocol stun low address 2 A7
```

Router B

```
stun peer-name 10.0.0.2
stun protocol-group 1 sdlc
stun protocol-group 2 sdlc
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 interface serial 1
!
interface serial 1
 ip address 10.0.0.2 255.0.0.0
 priority-group 1
!
interface serial 2
 no ip address
 encapsulation stun
 stun group 2
 stun route address A7 interface serial 1
!
priority-list 1 protocol stun high address 1 C1
priority-list 1 protocol stun low address 2 A7
```

SDLC Broadcast Example

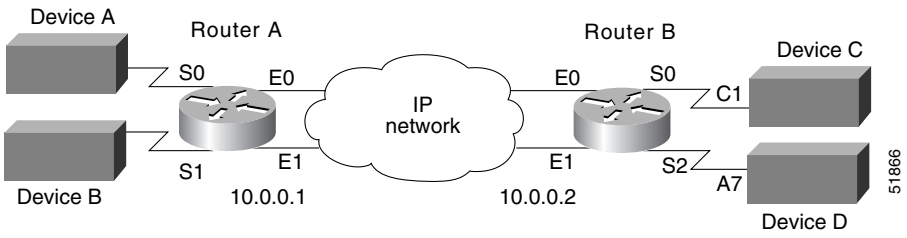
In the following example, an FEP views end stations 1, 2, and 3 as if they were on an SDLC multidrop link. Any broadcast frame sent from the FEP to Router A is duplicated and sent to each of the downstream routers (B and C.)

```
stun peer-name xxx.xxx.xxx.xxx
stun protocol-group 1 sdlc
interface serial 1
encapsulation stun
 stun group 1
 stun sdlc-role secondary
 sdlc virtual-multidrop
 sdlc address 1
 sdlc address 2
 sdlc address 3
 stun route address 1 tcp yyy.yyy.yyy.yyy local-ack
 stun route address 2 tcp zzz.zzz.zzz.zzz local-ack
 stun route address 3 tcp zzz.zzz.zzz.zzz local-ack
 stun route address FF tcp yyy.yyy.yyy.yyy
 stun route address FF tcp zzz.zzz.zzz.zzz
```

Serial Link Address Prioritization Using STUN TCP/IP Encapsulation Example

Assume that the link between Router A and Router B is a serial tunnel that uses the TCP/IP encapsulation as shown in Figure 152. Device A communicates with Device C (SDLC address C1) with a high priority. Device B communicates with Device D (SDLC address A7) with a normal priority. The configuration file for each router follows the figure.

Figure 152 STUN TCP/IP Encapsulation



Router A

```
stun peer-name 10.0.0.1
stun protocol-group 1 sdlc
stun protocol-group 2 sdlc
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 tcp 10.0.0.2 local-ack priority
 priority-group 1
!
interface serial 1
 no ip address
 encapsulation stun
 stun group 2
 stun route address A7 tcp 10.0.0.2 local-ack priority
 priority-group 2
!
interface ethernet 0
 ip address 10.0.0.1 255.0.0.0
 priority-group 3
!
interface ethernet 1
 ip address 10.0.0.3 255.0.0.0
priority-group 3
! This list tells interface Serial 0 which tcp port numbers on the WAN interface
! correspond to the high, medium, normal and low priority queues.
priority-list 1 protocol ip high tcp 1994
priority-list 1 protocol ip medium tcp 1990
priority-list 1 protocol ip normal tcp 1991
priority-list 1 protocol ip low tcp 1992
priority-list 1 protocol stun high address 1 C1


! This list tells interface Serial 1 which tcp port numbers
! on the WAN interface correspond to the high, medium, normal
! and low priority queues.
priority-list 2 protocol ip high tcp 1994
priority-list 2 protocol ip medium tcp 1990
priority-list 2 protocol ip normal tcp 1991
priority-list 2 protocol ip low tcp 1992
```

```
priority-list 2 protocol stun normal address 2 A7
! This list establishes the high, medium, normal, and low
! priority queues on the WAN interfaces.
priority-list 3 protocol ip high tcp 1994
priority-list 3 protocol ip medium tcp 1990
priority-list 3 protocol ip normal tcp 1991
priority-list 3 protocol ip low tcp 1992
!
hostname routerA
router igrp
network 1.0.0.0
```

Router B

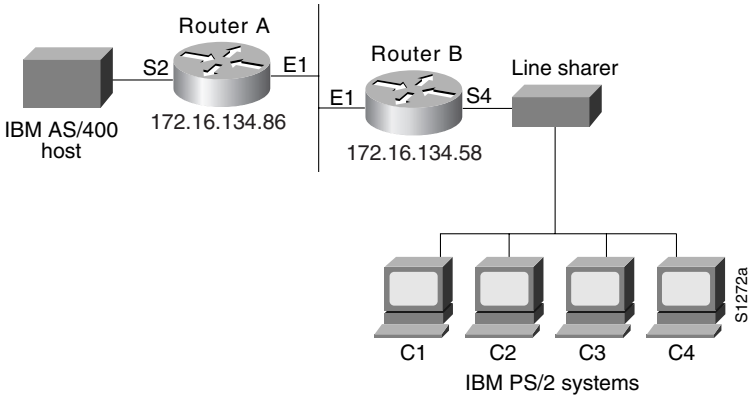
```
stun peer-name 10.0.0.2
stun protocol-group 1 sdlc
stun protocol-group 2 sdlc
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 tcp 10.0.0.1 local-ack priority
 priority-group 1
!
interface serial 2
 no ip address
 encapsulation stun
 stun group 2
 stun route address A7 tcp 10.0.0.1 local-ack priority
 priority-group 2
!
interface ethernet 0
 ip address 10.0.0.2 255.0.0.0
priority-group 3
!
interface ethernet 1
 ip address 10.0.0.4 255.0.0.0
priority-group 3
! This list tells interface Serial 0 which tcp port numbers
! on the WAN interface correspond to the high, medium, normal
! and low priority queues.
priority-list 1 protocol ip high tcp 1994
priority-list 1 protocol ip medium tcp 1990
priority-list 1 protocol ip normal tcp 1991
priority-list 1 protocol ip low tcp 1992
priority-list 1 protocol stun high address 1 C1


! This list tells interface Serial 2 which tcp port numbers
! on the WAN interface correspond to the high, medium, normal
! and low priority queues.
priority-list 2 protocol ip high tcp 1994
priority-list 2 protocol ip medium tcp 1990
priority-list 2 protocol ip normal tcp 1991
priority-list 2 protocol ip low tcp 1992
priority-list 2 protocol stun normal address 2 A7
! This list establishes the high, medium, normal, and low
! priority queues on the WAN interface(s).
priority-list 3 protocol ip high tcp 1994
priority-list 3 protocol ip medium tcp 1990
priority-list 3 protocol ip normal tcp 1991
priority-list 3 protocol ip low tcp 1992
!
```

```
hostname routerB
router igrp 109
network 1.0.0.0
```

STUN Multipoint Implementation Using a Line-Sharing Device Example

In Figure 153, four separate PS/2 computers are connected to a line-sharing device off of Router B. Each PS/2 computer has four sessions open on an AS/400 device attached to Router A. Router B functions as the primary station, while Router A functions as the secondary station. Both routers locally acknowledge packets from the IBM PS/2 systems.

Figure 153 STUN Communication Involving a Line-Sharing Device



The configuration file for the routers shown in Figure 153 follows.

Router A

```
! enter the address of the stun peer
stun peer-name 172.16.134.86
! specify that group 4 uses the SDLC protocol
stun protocol-group 4 sdlc
stun remote-peer-keepalive
!
interface ethernet 1
! enter the IP address for the Ethernet interface
 ip address 172.16.134.86 255.255.255.0
! description of IBM AS/400 link
interface serial 2
! description of IBM AS/400 link; disable the IP address on a serial interface
 no ip address
! enable STUN encapsulation on this interface
 encapsulation stun
! apply previously defined stun group 4 to serial interface 2
 stun group 4
! establish this router as a secondary station
stun sdlc-role secondary
! wait up to 63000 msec for a poll from the primary before timing out
 sdlc poll-wait-timeout 63000
! list addresses of secondary stations (PS/2 systems) attached to link
sdlc address C1
sdlc address C2
sdlc address C3
sdlc address C4
! use tcp encapsulation to send frames to SDLC stations C1, C2, C3, or
```

```
! C4 and locally terminate sessions with these stations
stun route address C1 tcp 172.16.134.58 local-ack
stun route address C2 tcp 172.16.134.58 local-ack
stun route address C3 tcp 172.16.134.58 local-ack
stun route address C4 tcp 172.16.134.58 local-ack
```

Router B

```
! enter the address of the stun peer
stun peer-name 172.16.134.58
! this router is part of SDLC group 4
stun protocol-group 4 sdlc
stun remote-peer-keepalive
!
interface ethernet 1
! enter the IP address for the Ethernet interface
ip address 172.16.134.58 255.255.255.0
!
! description of PS/2 link
interface serial 4
! disable the IP address on a serial interface
 no ip address
! enable STUN encapsulation on this interface
 encapsulation stun
! apply previously defined stun group 4 to serial interface 2
stun group 4
! establish this router as a primary station
stun sdlc-role primary
sdlc line-speed 9600
! wait 2000 milliseconds for a reply to a frame before resending it
sdlc t1 2000
! resend a frame up to four times if not acknowledged
sdlc n2 4


! list addresses of secondary stations (PS/2 systems) attached to link
sdlc address C1
sdlc address C2
sdlc address C3
sdlc address C4
! use tcp encapsulation to send frames to SDLC stations C1, C2, C3, or
! C4 and locally terminate sessions with these stations
stun route address C3 tcp 172.16.134.86 local-ack
stun route address C1 tcp 172.16.134.86 local-ack
stun route address C4 tcp 172.16.134.86 local-ack
stun route address C2 tcp 172.16.134.86 local-ack
! set the clock rate on this interface to 9600 bits per second
clock rate 9600
```

STUN Local Acknowledgment for SDLC Example

The following example shows a sample configuration for a pair of routers performing SDLC local acknowledgment.

Router A

```
stun peer-name 172.16.64.92
stun protocol-group 1 sdlc
stun remote-peer-keepalive
!
interface Serial 0
 no ip address
```

```
 encapsulation stun
 stun group 1
 stun sdlc-role secondary
 sdlc address C1
 stun route address C1 tcp 172.16.64.93 local-ack
 clock rate 19200
```

Router B

```
stun peer-name 172.16.64.93
stun protocol-group 1 sdlc
stun remote-peer-keepalive
!
interface Serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun sdlc-role primary
 sdlc line-speed 19200
 sdlc address C1
 stun route address C1 tcp 172.16.64.92 local-ack
 clock rate 19200
```

STUN Local Acknowledgment for Frame Relay Example

The following example describes an interface configuration for Frame Relay STUN with local acknowledgment:

```
stun peer-name 10.1.21.1 cls 4
stun protocol-group 120 sdlc
!
interface Serial1
 no ip address
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 22
!
interface Serial4
 no ip address
 encapsulation stun
 clock rate 9600
 stun group 120
 stun sdlc-role secondary
 sdlc address C1
 sdlc address C2
 stun route address C1 interface Serial1 dlci 22 04 local-ack
 stun route address C2 interface Serial1 dlci 22 08 local-ack
```

LOCADDR Priority Groups Example

The following example shows how to establish queueing priorities on a STUN interface based on an LU address:

```
stun peer-name 131.108.254.6
stun protocol-group 1 sdlc
! give locaddr-priority-list 1 a high priority for LU 02
locaddr-priority-list 1 02 high
! give locaddr-priority-list 1 a low priority for LU 05
locaddr-priority-list 1 05 low
!
```

```
interface serial 0
! disable the ip address for interface serial 0
 no ip address
! enable the interface for STUN
 encapsulation stun
 stun group 2
 stun route address 10 tcp 131.108.254.8 local-ack priority
! assign priority group 1 to the input side of interface serial 0
locaddr-priority 1
priority-group 1
```

LOCADDR Priority Groups for STUN Example

The following configuration example shows how to assign a priority group to an input interface:

Router A

```
stun peer-name 10.0.0.1
stun protocol-group 1 sdlc
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 tcp 10.0.0.2 local-ack priority
 clock rate 19200
 locaddr-priority 1
 priority-group 1
!
interface Ethernet 0
 ip address 10.0.0.1 255.255.255.0
!
 priority-list 1 protocol ip high tcp 1994
 priority-list 1 protocol ip medium tcp 1990
 priority-list 1 protocol ip normal tcp 1991
 priority-list 1 protocol ip low tcp 1992
```

Router B

```
stun peer-name 10.0.0.2
stun protocol-group 1 sdlc
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
interface serial 0
 no ip address
 encapsulation stun
 stun group 1
 stun route address C1 tcp 10.0.0.1 local-ack priority
 clock rate 19200
 locaddr-priority 1
 priority-group 1
!
interface Ethernet 0
 ip address 10.0.0.2 255.255.255.0
!
 priority-list 1 protocol ip high tcp 1994
 priority-list 1 protocol ip medium tcp 1990
 priority-list 1 protocol ip normal tcp 1991
 priority-list 1 protocol ip low tcp 1992
```

Block Serial Tunneling (BSTUN) Overview

This section describes how to configure BSTUN and contains the following sections:

Cisco's implementation of BSTUN provides the following features:

- Encapsulates Bisync, Adplex, ADT Security Systems, Inc., Diebold, and asynchronous generic traffic for transfer over router links. The tunneling of asynchronous security protocols (ASP) feature enables your Cisco 2500, 3600, 4000, 4500, or 7200 series router to support devices that use the following asynchronous security protocols:
  - adplex
  - adt-poll-select
  - adt-vari-poll
  - diebold
  - async-generic
  - mdi
- Provides a tunnel mechanism for BSTUN over Frame Relay, without using TCP/IP encapsulation.
- Supports Bisync devices and host applications without modification.
- Uses standard synchronous serial interfaces on Cisco 2500 series and the 4T network interface module (NIM) on the Cisco 4000 series and Cisco 4500 series.
- Supports point-to-point, multidrop, and virtual multidrop configurations.

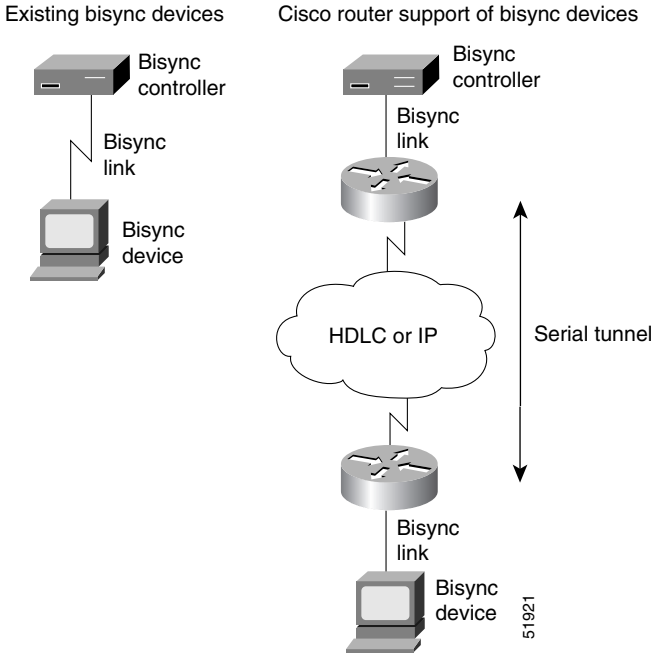
Note The async-generic item is not a protocol name. It is a command keyword used to indicate generic support of other asynchronous security protocols that are not explicitly supported.

Bisync Network Overview

The Bisync feature enables your Cisco 2500, 3600, 4000, 4500, 4700, and 7200 series routers to support devices that use the Bisync data-link protocol. This protocol enables enterprises to transport Bisync traffic over the same network that supports their SNA and multiprotocol traffic, eliminating the need for separate Bisync facilities.

At the access router, traffic from the attached Bisync device is encapsulated in IP. The Bisync traffic can then be routed across arbitrary media to the host site where another router supporting Bisync will remove the IP encapsulation headers and present the Bisync traffic to the Bisync host or controller over a serial connection. HDLC can be used as an alternative encapsulation method for point-to-point links. Figure 154 shows how you can reconfigure an existing Bisync link between two devices and provide the same logical link without any changes to the existing Bisync devices.

Figure 154 Routers Consolidating Bisync Traffic by Encapsulation in IP or HDLC



The routers transport all Bisync blocks between the two devices in pass-through mode using BSTUN as encapsulation. BSTUN uses the same encapsulation architecture as STUN, but is implemented on an independent tunnel.

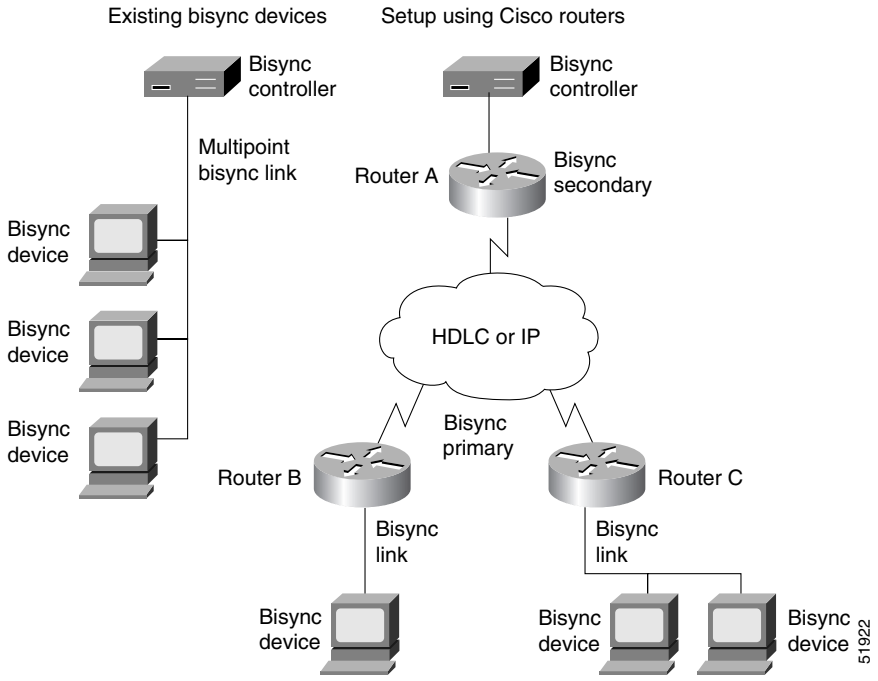
Point-to-Point and Multidrop Support

The Bisync feature supports point-to-point, multidrop, and virtual multidrop Bisync configurations.

In point-to-point operation, the Bisync blocks between the two point-to-point devices are received and forwarded transparently by the Cisco IOS software. The contention to acquire the line is handled by the devices themselves.

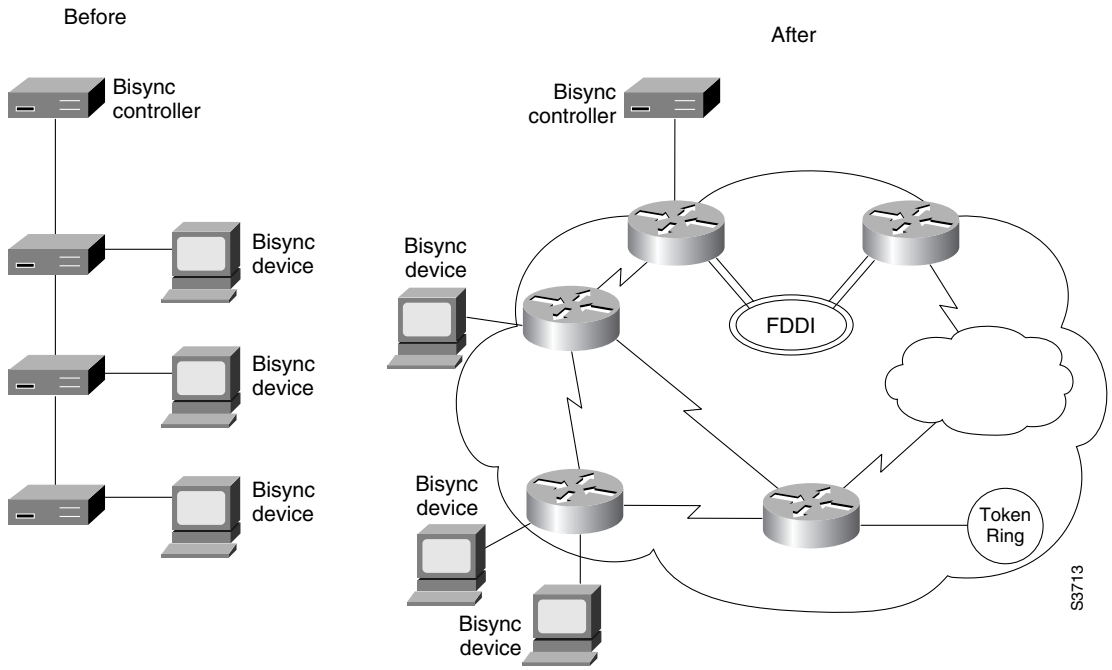
Cisco's Bisync multipoint operation is provided as a logical multipoint configuration. Figure 155 shows how a multipoint Bisync link is reconfigured using Cisco routers. Router A is configured as Bisync secondary. It monitors the address field of the polling or selection block and uses this address information to put into the BSTUN frame for BSTUN to deliver to the correct destination router. To simulate the Bisync multidrop, an EOT block is sent by the Bisync primary router before a poll or selection block. This ensures that Bisync tributary stations are in control mode before being polled or selected.

Figure 155 Multipoint Bisync Link Reconfigured Using Routers



Multidrop configurations are common in Bisync networks where up to 8 or 10 Bisync devices are frequently connected to a Bisync controller port over a single low-speed link. Bisync devices from different physical locations in the network appear as a single multidrop line to the Bisync host or controller. Figure 156 illustrates a multidrop Bisync configuration before and after implementing routers.

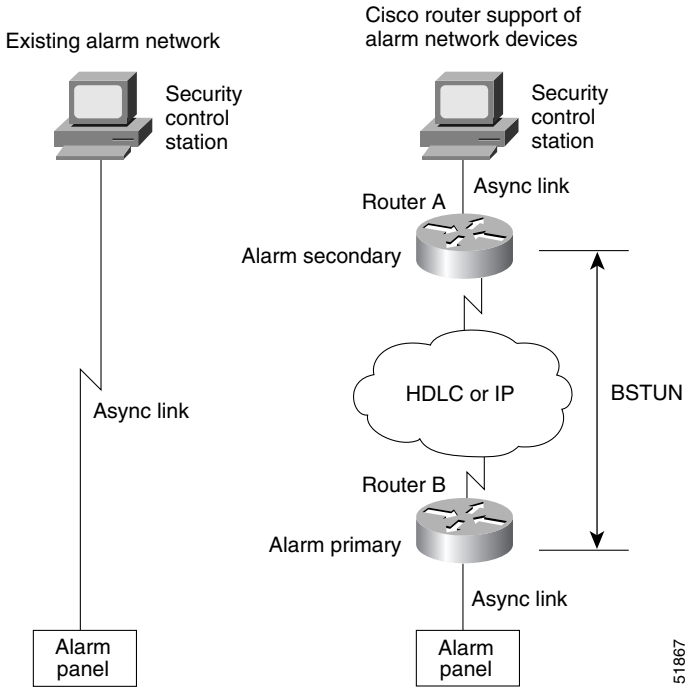
Figure 156 Integrating Bisync Devices over a Multiprotocol Network



Asynchronous Network Overview

These protocols enable enterprises to transport polled asynchronous traffic over the same network that supports their SNA and multiprotocol traffic, eliminating the need for separate facilities. Figure 157 shows how you can reconfigure an existing asynchronous link between two security devices and provide the same logical link without any changes to the existing devices.

Figure 157 Routers Consolidate Polled Asynchronous Traffic Using Encapsulation in IP or HDLC



Router A is configured as the secondary end of the BSTUN asynchronous link and is attached to the security control station; Router B is configured as the primary end of the BSTUN asynchronous link and has one or more alarm panels attached to it.

At the downstream router, traffic from the attached alarm panels is encapsulated in IP. The asynchronous (alarm) traffic can be routed across arbitrary media to the host site where the upstream router supporting these protocols removes the IP encapsulation headers and presents the original traffic to the security control station over a serial connection. High-Level Data Link Control (HDLC) can be used as an alternative encapsulation method for point-to-point links.

The routers transport all asynchronous (alarm) blocks between the two devices in passthrough mode using BSTUN for encapsulation. BSTUN uses the same encapsulation architecture as STUN, but is implemented on an independent tunnel. As each asynchronous frame is received from the line, a BSTUN header is added to create a BSTUN frame, and then BSTUN is used to deliver the frame to the correct destination router.

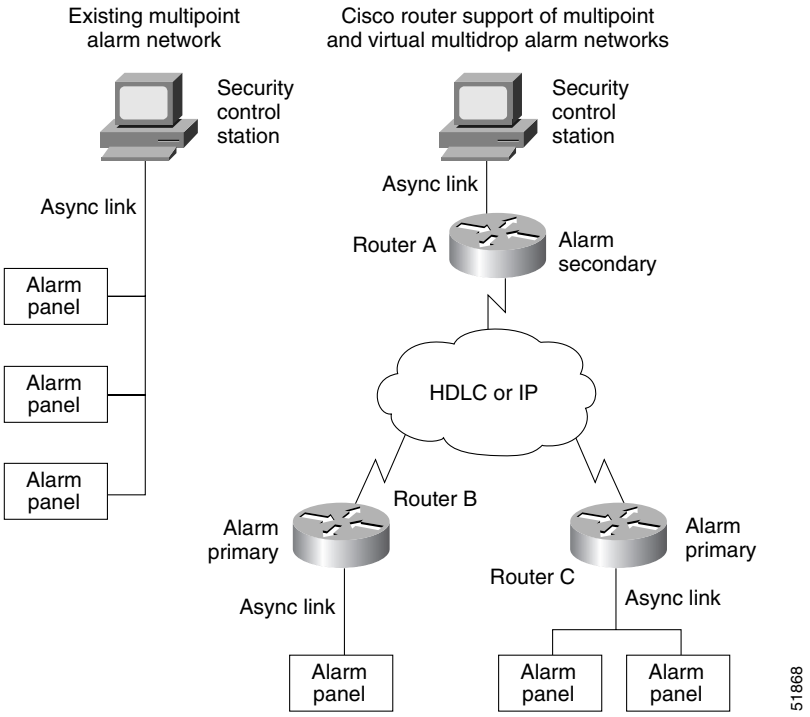
The Cisco routers do not perform any local acknowledgment or cyclic redundancy check (CRC) calculations on the asynchronous alarm blocks. The two end devices are responsible for error recovery in the asynchronous alarm protocol.

Virtual Multidrop Support for Multipoint Security Network Configurations

Multipoint configurations are common in security networks, where a number of alarm panels are frequently connected to a security control station over a single low-speed link. Our virtual multidrop support allows alarm panels from different physical locations in the network to appear as a single multidrop line to the security control station. Both Adplex and ADT are virtual multidropped protocols.

Multidrop operation is provided as a logical multipoint configuration. Figure 158 shows how a multipoint security network is reconfigured using Cisco routers. Router A is configured as an alarm secondary node, routers B and C are configured as alarm primary nodes. Router A monitors the address field of the polling or selection block and puts this address information in the BSTUN frame so BSTUN can deliver the frame to the correct downstream node.

Figure 158 Multipoint Asynchronous Security Protocol Link Reconfigured Using Routers



Frame Sequencing

Both Bisync and asynchronous alarm protocols are half-duplex protocols; data can be sent in either direction, but only in one direction at a time. Each block sent is acknowledged explicitly by the remote end. To avoid the problem associated with simultaneous sending of data, there is an implicit role of primary and secondary station.

Frame Sequencing in Bisync Networks

In a multidrop setup in Bisync networks, the Bisync control station is primary and the tributary stations are secondary. In a point-to-point configuration, the primary role is assumed by the Bisync device that has successfully acquired the line for sending data through the ENQ bidding sequence. The primary role stays with this station until it sends EOT.

To protect against occasional network latency, which causes the primary station to time out and resend the block before the Bisync block sent by the secondary is received, the control byte of the encapsulating frame is used as a sequence number. This sequence number is controlled and monitored by the primary Bisync router. This allows the primary Bisync router to detect and discard "late" Bisync blocks sent by the secondary router and ensure integrity of the Bisync link.

Note Frame sequencing is implemented in passthrough mode only.

Frame Sequencing in Asynchronous Networks

Network delays in asynchronous networks make it possible for a frame to arrive "late," meaning that the poll-cycling mechanism at the security control station has already moved on to poll the next alarm panel in sequence when it receives the poll response from the previous alarm panel.

To protect against this situation, routers configured for adplex or for adt-poll-select protocols use a sequence number built into the encapsulating frame to detect and discard late frames. The "upstream" router (connected to the security control station) inserts a frame sequence number into the protocol header, which is shipped through the BSTUN tunnel and bounced back by the "downstream" router (connected to the alarm panel). The upstream router maintains a frame-sequence count for the line, and checks the incoming frame-sequence number from the downstream router. If the two frame-sequence numbers do not agree, the frame is considered late (out of sequence) and is discarded.

Because the adt-vari-poll option allows the sending of unsolicited messages from the alarm panel, frame sequencing is not supported for this protocol.

Note Polled asynchronous (alarm) protocols are implemented only in passthrough mode. There is no support for local acknowledgment.

BSTUN Configuration Task List

The Bisync feature is configured similar to SDLC STUN, but is configured as a protocol within a BSTUN feature. To configure and monitor Bisync with BSTUN, perform the tasks in the following sections:

- Configuring Direct Serial Encapsulation for Passthrough Peers, page 36
- Configuring Local Acknowledgment Peers, page 36

The "BSTUN Configuration Examples" section on page 36 follows these tasks.

Enabling BSTUN

To enable BSTUN in IP networks, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# bstun peer-name ip-address`	Enables BSTUN.
Step 2	`Router(config)# bstun lisnsap sap-value`	Configures a SAP on which to listen for incoming calls.

The IP address in the **bstun peer-name** command defines the address by which this BSTUN peer is known to other BSTUN peers that are using the TCP transport. If this command is unconfigured or the **no** form of this command is specified, all BSTUN routing commands with IP addresses are deleted. BSTUN routing commands without IP addresses are not affected by this command.

The **bstun lisnsap** command specifies a SAP on which to detect incoming calls.

Defining the Protocol Group

Define a BSTUN group and specify the protocol it uses. To define the protocol group, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun protocol-group group-number {bsc | bsc-local-ack | adplex | adt-poll | adt-poll-select | adt-vari-poll | diebold | async-generic | mdi}`	Defines the protocol group.

The **bsc-local-ack** protocol option only works for 3270 Bisync uses.

The block serial protocols include bsc, bsc-local-ack, adplex, adt-poll-select, adt-vari-poll, diebold, async-generic, and mdi.

Traditionally, the adt-poll-select protocol is used over land-based links, while the adt-vari-poll protocol is used over satellite (VSAT) links. The adt-vari-poll protocol typically uses a much slower polling rate when alarm consoles poll alarm panels because adt-vari-poll allows alarm panels to send unsolicited messages to the alarm console. In an adt-vari-poll configuration, alarm panels do not have wait for the console to poll them before responding with an alarm, they automatically send the alarm.

Interfaces configured to run the adplex protocol have their baud rate set to 4800 bps, use even parity, 8 data bits, 1 start bit, and 1 stop bit.

Interfaces configured to run the adt-poll-select and adt-vari-poll protocols have their baud rate set to 600 bps, use even parity, 8 data bits, 1 start bit, and 1.5 stop bits. If different line configurations are required, use the **rxspeed**, **txspeed**, **databits**, **stopbits**, and **parity** line configuration commands to change the line attributes.

Interfaces configured to run the diebold protocol have their baud rate set to 300 bps, use even parity, 8 data bits, 1 start bit, and 2 stop bits. If different line configurations are required, use the **rxspeed**, **txspeed**, **databits**, and **parity** line configuration commands to change the line attributes.

Interfaces configured to run the async-generic protocol have their baud rate set to 9600 bps, use no parity, 8 data bits, 1 start bit, and 1 stop bit. If different line configurations are required, use the **rxspeed**, **txspeed**, **databits**, **stopbits**, and **parity** line configuration commands to change the line attributes.

Interfaces configured to run the mdi protocol have their baud rate set to 600 bps, use even parity, 8 data bits, 1 start bit, and 1.5 stop bits. If different line configurations are required, use the **rxspeed**, **txspeed**, **databits**, **stopbits**, and **parity** line configuration commands to change the line attributes. The mdi protocol allows alarm panels to be sent to the MDI alarm console.

Enabling BSTUN Keepalive

To define the number of times to attempt a peer connection before declaring the peer connection be down, use the following command in global configuration mode:

Command	Purpose
`Router(config)# bstun keepalive-count`	Specifies the number of times to attempt a peer connection.

Enabling BSTUN Remote Keepalive

To enable detection of the loss of a peer, use the following command in global configuration mode:

:

Command	Purpose
`Router(config)# bstun remote-peer-keepalive` *seconds*	Enables detection of the loss of a peer.

Enabling Frame Relay Encapsulation

To enable Frame Relay encapsulation, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface serial` *number*	Specifies a serial port.
Step 2	`Router(config)# encapsulation frame-relay`	Enables Frame Relay encapsulation on the serial port.

Defining Mapping Between BSTUN and DLCI

To configure the mapping between BSTUN and the DLCI, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# frame-relay map bstun dlci`	Defines the mapping between BSTUN and the DLCI when using BSC passthrough.
`Router(config-if)# frame-relay map llc2 dlci`	Defines the mapping between BSTUN and the DLCI when using BSC local acknowledgment.

Note Direct encapsulation over Frame Relay is supported only for an encapsulation type of cisco, configured using the **encapsulation frame-relay** command.

Configuring BSTUN on the Serial Interface

Configure BSTUN on the serial interface before issuing any further BSTUN or protocol configuration commands for the interface. To configure the BSTUN function on a specified interface, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# interface serial number`	Specifies a serial port.
Step 2	`Router(config-if)# encapsulation bstun`	Configures BSTUN on an interface.

Note Configure the encapsulation bstun command on an interface before configuring any other BSTUN commands for the interface.

Placing a Serial Interface in a BSTUN Group

Each BSTUN-enabled interface on a router must be placed in a previously defined BSTUN group. Packets will only travel between BSTUN-enabled interfaces that are in the same group. To assign a serial interface to a BSTUN group, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# bstun group group-number`	Assigns a serial interface to a BSTUN group.

Specifying How Frames Are Forwarded

To specify how frames are forwarded when received on a BSTUN interface, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bstun route address address-number interface serial number`	Propagates the serial frame that contains a specific address. HDLC encapsulation is used to propagate the serial frames.
`Router(config-if)# bstun route all interface serial number`	Propagates all BSTUN traffic received on the input interface, regardless of the address contained in the serial frame. HDLC encapsulation is used to propagate the serial frames.
`Router(config-if)# bstun route address address-number tcp ip-address`	Propagates the serial frame that contains a specific address. TCP encapsulation is used to propagate frames that match the entry.
`Router(config-if)# bstun route all tcp ip-address`[1]	Propagates all BSTUN traffic received on the input interface, regardless of the address contained in the serial frame. TCP encapsulation is used to propagate frames that match the entry.
`Router(config-if)# bstun route address cu-address interface serial serial-int [dlci dlci]`	Propagates the serial frame that contains a specific address. Specifies the control unit address for the Bisync end station. Frame Relay encapsulation is used to propagate the serial frames.
`Router(config-if)# bstun route all interface serial serial-int [dlci dlci]`	Propagates all frames regardless of the control unit address for the Bisync end station. Frame Relay encapsulation is used to propagate the serial frames in bisync passthrough mode.
`Router(config-if)# bstun route address cu-address interface serial serial-int [dlci dlci rsap] [priority priority]`	Propagates the serial frame that contains a specific address. Specifies the control unit address for the bisync end station. Frame Relay encapsulation is used to propagate the serial frames for Bisync local acknowledgment mode.
`Router(config-if)# bstun route all interface serial serial-int [dlci dlci rsap] [priority priority]`	Propagates all BSTUN traffic received on the input interface, regardless of the address contained in the serial frame. Frame Relay encapsulation is used to propagate the serial frames.

1. The **bstun route all tcp** command functions in either passthrough or local acknowledgment mode.

Note Every BSTUN route statement must have a corresponding route statement on the BSTUN peer. For example, a **bstun route address** *address1* **tcp** *peer2ip* statement on PEER1 must have a corresponding **bstun route address** *address1* **tcp** *peer1ip* statement on PEER2. Similarly, a **bstun route address** statement cannot map to a **bstun route all** statement, and vice versa.

For Bisync local acknowledgment, we recommend that you use the **bstun route all tcp** command. This command reduces the amount of duplicate configuration detail that would otherwise be needed to specify devices at each end of the tunnel.

Setting Up BSTUN Traffic Priorities

You can assign BSTUN traffic priorities based on either the BSTUN header or the TCP port. To prioritize traffic, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# priority-list list-number protocol bstun queue [gt packetsize] [lt packetsize] address bstun-group bsc-addr`	Establishes BSTUN queueing priorities based on the BSTUN header.
`Router(config)# priority-list list-number protocol ip queue tcp tcp-port-number`	Assigns a queueing priority to TCP port.

You can customize BSTUN queueing priorities based on either the BSTUN header or TCP port. To customize priorities, use one of the following commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# queue-list list-number protocol bstun queue [gt packetsize][lt packetsize] address bstun-group bsc-addr`	Customizes BSTUN queueing priorities based on the BSTUN header.
`Router(config)# queue-list list-number protocol ip queue tcp tcp-port-number`	Customizes BSTUN queueing priorities based on the TCP port.

Note Because the asynchronous security protocols share the same tunnels with Bisync when configured on the same routers, any traffic priorities configured for the tunnel apply to both Bisync and the various asynchronous security protocols.

Configuring Protocol Group Options on a Serial Interface

Depending on the selected block serial protocol group, you must configure one or more options for that protocol group. The options for each of these protocol groups are explained in the following sections:

- Configuring Bisync Options on a Serial Interface, page 34
- Configuring Asynchronous Security Protocol Options on a Serial Interface, page 35

Configuring Bisync Options on a Serial Interface

To configure Bisync options on a serial interface, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# bsc char-set {ascii | ebcdic}`	Specifies the character set used by the Bisync support feature.
`Router(config-if)# bsc contention address`	Specifies an address on a contention interface.

Command	Purpose
`Router(config-if)# bsc dial-contention time-out`	Specifies that the router at the central site will behave as a central router with dynamic allocation of serial interfaces. The timeout value is the length of time an interface can be idle before it is returned to the idle interface pool.
`Router(config-if)# bsc extended-address poll-address select-address`	Specifies a nonstandard Bisync address.
`Router(config-if)# full-duplex`	Specifies that the interface can run Bisync in full-duplex mode.
`Router(config-if)# bsc pause time`	Specifies the amount of time between the start of one polling cycle and the next.
`Router(config-if)# bsc poll-timeout time`	Specifies the timeout for a poll or a select sequence.
`Router(config-if)# bsc host-timeout time`	Specifies the timeout for a nonreception of poll or a select sequence from the host. If the frame is not received within this time, the remote connection will be deactivated.
`Router(config-if)# bsc primary`	Specifies that the router is acting as the primary end of the Bisync link.
`Router(config-if)# bsc retries retry-count`	Specifies the number of retries before a device is considered to have failed.
`Router(config-if)# bsc secondary`	Specifies that the router is acting as the secondary end of the Bisync link.
`Router(config-if)# bsc spec-poll`	Specifies specific polls, rather than general polls, used on the host-to-router connection.
`Router(config-if)# bsc servlim servlim-count`	Specifies the number of cycles of the active poll list that are performed between polls to control units in the inactive poll list.

Configuring Asynchronous Security Protocol Options on a Serial Interface

To configure asynchronous security protocol options on a serial interface, use one or more of the following commands in interface configuration mode:

Command	Purpose
`Router(config-if)# asp role primary`	Specifies that the router is acting as the primary end of the polled asynchronous link.
`Router(config-if)# asp role secondary`	Specifies that the router is acting as the secondary end of the polled asynchronous link.
`Router(config-if)# asp addr-offset address-offset`	Configures an asynchronous port to send and receive polled asynchronous traffic through a BSTUN tunnel.
`Router(config-if)# asp rx-ift interframe-timeout`	For asynchronous-generic configurations, specifies the timeout period between frames to delineate the end of one frame being received from the start of the next frame.

Configuring Direct Serial Encapsulation for Passthrough Peers

To configure direct serial encapsulation for passthrough peers, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# frame relay map bstun`	Configures the Frame Relay interface for passthrough.

Configuring Local Acknowledgment Peers

To configure local acknowledgment peers, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# frame-relay map llc2 dlci`	Configures the Frame Relay interface for local acknowledgment.

Monitoring and Maintaining the Status of BSTUN

To list statistics for BSTUN interfaces, protocol groups, number of packets sent and received, local acknowledgment states, and other activity information, use the following commands in EXEC mode:

Command	Purpose
`Router# show bstun [group bstun-group-number] [address address-list]`	Lists the status display fields for BSTUN interfaces.
`Router# show bsc [group bstun-group-number] [address address-list]`	Displays status of the interfaces on which Bisync is configured.

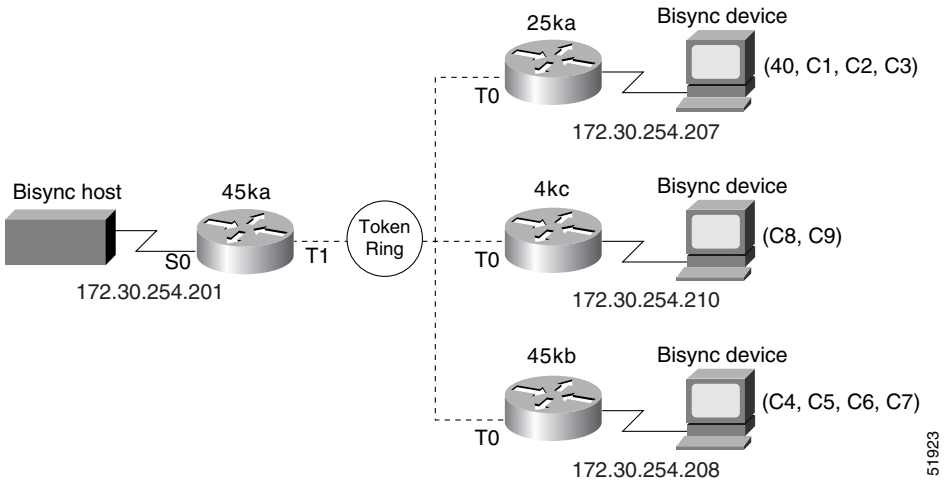
BSTUN Configuration Examples

The following sections provide BSTUN configuration examples:

Simple Bisync Configuration Example

Figure 159 shows a simple Bisync configuration example.

Figure 159 Simple Bisync Configuration



The configuration files for the routers shown in Figure 159 follows.

Router 45ka

```
version 10.2
!
hostname 45ka
!
no ip domain-lookup
!
bstun peer-name 172.30.254.201
bstun protocol-group 1 bsc
!
interface ethernet 0
 ip address 198.92.0.201 255.255.255.0
 media-type 10BaseT
!
interface ethernet 1
 no ip address
 shutdown
 media-type 10BaseT
!
```

```
interface serial 0
 no ip address
 encapsulation bstun
 clock rate 19200
 bstun group 1
 bsc char-set ebcdic
 bsc secondary
 bstun route address C9 tcp 172.30.254.210
bstun route address C8 tcp 172.30.254.210
bstun route address C7 tcp 172.30.254.208
bstun route address C6 tcp 172.30.254.208
bstun route address C5 tcp 172.30.254.208
bstun route address C4 tcp 172.30.254.208
bstun route address C3 tcp 172.30.254.207
bstun route address C2 tcp 172.30.254.207
bstun route address C1 tcp 172.30.254.207
bstun route address 40 tcp 172.30.254.207
!
interface serial 1
no ip address
shutdown
!
interface serial 2
 no ip address
 shutdown
!
interface serial 3
 no ip address
 shutdown
!
interface tokenring 0
 no ip address
 shutdown
!
interface tokenring 1
 ip address 172.30.254.201 255.255.255.0
 ring-speed 16
!
line con 0
line aux 0
line vty 0 4
login
!
end
```

Router 25ka

```
version 10.2
!
hostname 25ka
!
no ip domain-lookup
!
bstun peer-name 172.30.254.207
bstun protocol-group 1 bsc
!
interface serial 0
 no ip address
 shutdown
!
```

```
interface serial 1
 no ip address
 encapsulation bstun
 clock rate 19200
 bstun group 1
 bsc char-set ebcdic
 bsc primary
 bstun route address C3 tcp 172.30.254.201
 bstun route address C2 tcp 172.30.254.201
 bstun route address C1 tcp 172.30.254.201
 bstun route address 40 tcp 172.30.254.201
!
interface tokenring 0
 ip address 172.30.254.207 255.255.255.0
 ring-speed 16
!
interface bri 0
 no ip address
 shutdown
!
line con 0
line aux 0
line vty 0 4
login
!
end
```

Configuration for Router 4kc

```
version 10.2
!
hostname 4kc
!
no ip domain-lookup
!
bstun peer-name 172.30.254.210
bstun protocol-group 1 bsc
!
interface ethernet 0
 ip address 198.92.0.210 255.255.255.0
 media-type 10BaseT
!
interface serial 0
 no ip address
 encapsulation bstun
 clock rate 19200
 bstun group 1
 bsc char-set ebcdic
 bsc primary
 bstun route address C9 tcp 172.30.254.201
 bstun route address C8 tcp 172.30.254.201
!
interface serial 1
 no ip address
 shutdown
!
interface serial 2
 no ip address
 shutdown
!


interface serial 3
 no ip address
```

```
 shutdown
!
interface tokenring 0
 ip address 172.30.254.210 255.255.255.0
 ring-speed 16
!
interface tokenring 1
 no ip address
 shutdown!
line con 0
line aux 0
line vty 0 4
login
!
end
```

Router 25kb

```
version 10.2
!
hostname 25kb
!
no ip domain-lookup
!
bstun peer-name 172.30.254.208
bstun protocol-group 1 bsc
!
interface serial 0
 no ip address
 encapsulation bstun
 no keepalive
 clock rate 19200
 bstun group 1
 bsc char-set ebcdic
 bsc primary
 bstun route address C7 tcp 172.30.254.201
 bstun route address C6 tcp 172.30.254.201
 bstun route address C5 tcp 172.30.254.201
 bstun route address C4 tcp 172.30.254.201
!
interface serial 1
 no ip address
 shutdown
!
interface tokenring 0
 ip address 172.30.254.208 255.255.255.0
 ring-speed 16
!
!
line con 0
line aux 0
line vty 0 4
login
!
end
```

Bisync Addressing on Contention Interfaces Example

The following examples show user-configurable addressing on contention interfaces:

Remote Devices

```
bstun peer-name 1.1.1.20
bstun protocol-group 1 bsc
interface serial 0
 bstun group 1
 bsc contention 20
 bstun route address 20 tcp 1.1.1.1
```

Host Device

```
bstun peer-name 1.1.1.1
bstun protocol-group 1 bsc
interface serial 0
 bstun group 1
 bsc dial-contention 100
 bstun route address 20 tcp 1.1.1.20
 bstun route address 21 tcp 1.1.1.21
```

Nonstandard Bisync Addressing Example

This example specifies an extended address on serial interface 0:

```
bstun peer-name 1.1.1.1
bstun protocol-group 1 bsc
!
interface serial 0
 bstun group 1
 bsc extended-address 23 83
 bsc extended-address 87 42
 bsc primary
 bstun route address 23 tcp 1.1.1.20
```

Priority Queueing: With Priority Based on BSTUN Header Example

In the following example, the output interface examines header info and places packets with the BSTUN header on specified output queue:

```
priority-list 1 protocol bstun normal
interface serial 0
 priority-group 1
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route all interface serial 0
                ...or...
bstun route address C1 interface serial 0
```

Priority Queueing: With Priority Based on BSTUN Header and Packet Sizes Example

In the following example, the output interface examines header information and packet size and places packets with the BSTUN header that match criteria (gt or lt specified packet size) on specified output queue:

```
priority-list 1 protocol bstun low gt 1500
priority-list 1 protocol bstun hi lt 500
interface serial 0
 priority-group 1
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route all interface serial 0
                ...or...
bstun route address C1 interface serial 0
```

Priority Queueing: With Priority Based on BSTUN Header and Bisync Address Example

In the following example, the output interface examines header information and Bisync address and places packets with the BSTUN header that match Bisync address on specified output queue:

```
priority-list 1 protocol bstun normal address 1 C1
interface serial 0
 priority-group 1
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route address C1 interface serial 0
```

Priority Queueing: With Priority Based on BSTUN TCP Ports Example

In the following example, the output interface examines TCP port number and places packets with the BSTUN port number (1976) on specified output queue:

```
priority-list 1 protocol ip high tcp 1976
interface serial 0
 priority-group 1
interface serial 1
 encapsulation bstun
 bstun group 1
 bstun route all tcp 200.190.30.1
```

Priority Queueing: With Priority Based on BSTUN TCP Ports and Bisync Address Example

In the following example, four TCP/IP sessions (high, medium, normal, and low) are established with BSTUN peers using BSTUN port numbers. The input interface examines the Bisync address and uses the specified output queue definition to determine which BSTUN TCP session to use for sending the packet to the BSTUN peer.

The output interface examines the TCP port number and places packets with the BSTUN port numbers on the specified output queue.

```
priority-list 1 protocol ip high   tcp 1976
priority-list 1 protocol ip medium tcp 1977
priority-list 1 protocol ip normal tcp 1978
priority-list 1 protocol ip low    tcp 1979
!
priority-list 1 protocol bstun normal address 1 C1
!
interface serial 0
 priority-group 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route address C1 tcp 200.190.30.1 priority
 priority-group 1
```

Custom Queueing: With Priority Based on BSTUN Header Example

In the following example, the output interface examines header info and places packets with the BSTUN header on specified output queue.

```
queue-list 1 protocol bstun normal
!
interface serial 0
 custom-queue-list 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bstun route all interface serial 0
```

Custom Queueing: With Priority Based on BSTUN Header and Packet Size Example

In the following example, the output interface examines header information and packet size and places packets with the BSTUN header that match criteria (gt or lt specified packet size) on specified output queue.

```
queue-list 1 protocol bstun low gt 1500
queue-list 1 protocol bstun high lt 500
!
interface serial 0
 custom-queue-list 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bstun route all interface serial 0
```

Custom Queueing: With Priority Based on BSTUN Header and Bisync Address Example

In the following example, the output interface examines header info and Bisync address and places packets with the BSTUN header that match Bisync address on specified output queue.

```
queue-list 1 protocol bstun normal address 1 C1
!
interface serial 0
 custom-queue-list 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route address C1 interface serial 0
```

Custom Queueing: With Priority Based on BSTUN TCP Ports Example

In the following example, the output interface examines the TCP port number and places packets with the BSTUN port number (1976) on specified output queue:

```
queue-list 1 protocol ip high tcp 1976
!
interface serial 0
 custom-queue-list 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bstun route all tcp 200.190.30.1
```

Custom Queueing: With Priority Based on BSTUN TCP Ports and Bisync Address Example

In the following example, four TCP/IP sessions (high, medium, normal, and low) are established with BSTUN peers using BSTUN port numbers. The input interface examines the Bisync address and uses the specified output queue definition to determine which BSTUN TCP session to use.

The output interface examines the TCP port number and places packets with the BSTUN port numbers on the specified output queue.

For Bisync addressing, output queues map as shown in Table 5:

Table 5 Bisync Addressing Output Queues

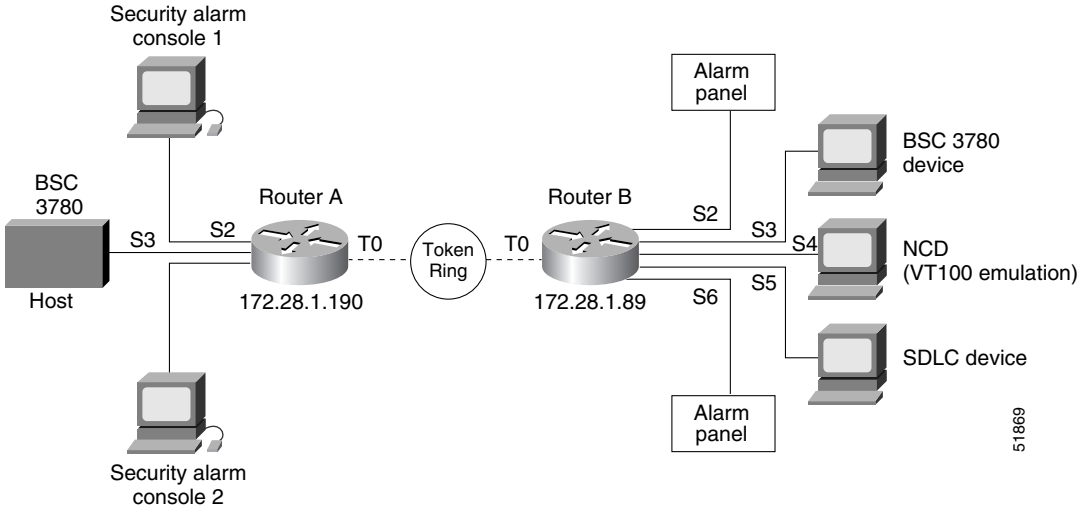
Output Queue	Session Mapped	BSTUN Port
1	Medium	1977
2	Normal	1978
3	Low	1979
4–10	High	1976

```
queue-list 1 protocol ip high tcp 1976
queue-list 1 protocol ip medium tcp 1977
queue-list 1 protocol ip normal tcp 1978
queue-list 1 protocol ip low tcp 1979
!
priority-list 1 protocol bstun normal address 1 C1
!
interface serial 0
 custom-queue-list 1
!
interface serial 1
 encapsulation bstun
 bstun group 1
 bsc char-set ebcdic
 bstun route address C1 tcp 200.190.30.1 priority
 custom-queue-list 1
```

Asynchronous Configuration Example

In the following example, Router A and Router B are configured for both Adplex and Bisync across the same BSTUN as shown in Figure 160.

Figure 160 Combined Adplex and Bisync Configuration Example



Router A

```
version 11.0
!
hostname router-a
!
bstun peer-name 172.28.1.190
bstun protocol-group 1 bsc
bstun protocol-group 2 adplex
bstun protocol-group 3 adplex
!
interface serial 0
 no ip address
!
interface serial 1
 no ip address
!
interface serial 2
 physical-layer async
 description Connection to 1st Security Alarm Console.
 no ip address
 encapsulation bstun
 no keepalive
 bstun group 2
 bstun route address 2 tcp 172.28.1.189
 bstun route address 3 tcp 172.28.1.189
 adplex secondary
!


interface serial 3
 description Connection to BSC 3780 host.
 no ip address
 encapsulation bstun
```

```
 no keepalive
 clock rate 9600
 bstun group 1
 bstun route all tcp 172.28.1.189
 bsc char-set ebcdic
 bsc contention
!
interface serial 4
 physical-layer async
 description Connection to 2nd Security Alarm Console.
 no ip address
 encapsulation bstun
 no keepalive
 bstun group 3
 bstun route address 2 tcp 172.28.1.189
 bstun route address 3 tcp 172.28.1.189
 adplex secondary
!
interface serial 5
 no ip address
!
interface serial 6
 no ip address
!
interface serial 7
 no ip address
!
interface serial 8
 no ip address
!
interface serial 9
 no ip address
!
interface tokenring 0
 ip address 172.28.1.190 255.255.255.192
 ring-speed 16
!
interface BRI0
 ip address
 shutdown
!
ip host ss10 172.28.0.40
ip host s2000 172.31.0.2
ip route 0.0.0.0 0.0.0.0 172.28.1.129
!
snmp-server community public RO
!
line con 0
 exec-timeout 0 0
line 2
 no activation-character
 transport input-all
 parity even
 stopbits 1
 rxspeed 4800
 txspeed 4800
line 4
 transport input all
 parity even
 stopbits 1
 rxspeed 4800
 txspeed 4800
line aux 0
 transport input all
```

```
line vty 0 4
 password mango
 login
!
end
```

Router B

```
version 11.0
!
hostname router-b
!
bstun peer-name 172.28.1.189
bstun protocol-group 1 bsc
bstun protocol-group 2 adplex
bstun protocol-group 3 adplex
source-bridge ring-group 100
!
interface serial 0
 no ip address
!
interface serial 1
 no ip address
!
interface serial 2
 physical-layer async
 description Connection to Security Alarm Panel.
 no ip address
 encapsulation bstun
 no keepalive
 bstun group 2
 bstun route all tcp 172.28.1.190
 adplex primary
!
interface serial 3
 description Connection to BSC 3780 device.
 no ip address
 encapsulation bstun
 no keepalive
 clock rate 9600
 bstun group 1
 bstun route all tcp 172.28.1.190
 bsc char-set ebcdic
 bsc contention
!
interface serial 4
 physical-layer async
 description Connection to async port on NCD (VT100 terminal emulation).
 no ip address
!


interface serial 5
 no ip address
 encapsulation sdlc-primary
 no keepalive
 nrzi-encoding
 clock rate 9600
 sdllc traddr 4000.0000.4100 222 2 100
 sdlc address C1
 sdllc xid C1 05D40003
 sdllc partner 4000.0000.0307 C1
!
interface serial 6
```

```
 description Connection to alarm panel.
 physical-layer async
 no ip address
 encapsulation bstun
 no keepalive
 bstun group 3
 bstun route all tcp 172.28.1.190
 adplex primary
!interface serial 7
 no ip address
!
interface serial 8
 no ip address
!
interface serial 9
 no ip address
!
interface tokenring 0
 ip address 172.28.1.189 255.255.255.192
 ring-speed 16
 source-bridge 4 1 100
!
interface BRI0
 ip address
 shutdown
!
ip host ss10 172.28.0.40
ip host s2000 172.31.0.2
ip route 0.0.0.0 0.0.0.0 172.28.1.129
!
snmp-server community public RO
!
line con 0
 exec-timeout 0 0
line 2
 no activation-character
 transport input-all
 parity even
 stopbits 1
 rxspeed 4800
 txspeed 4800
line 4
 transport input all
 stopbits 1
line 6
 transport input all
 parity even
 stopbits 1
 rxspeed 4800
 txspeed 4800
line 7
 transport input all
line aux 0
 transport input all
line vty 0 4
 password mango
 login
!
end
```

BSTUN-over-Frame Relay Configuration with Local Acknowledgment Example

The following example configures BSTUN over Frame Relay with local acknowledgment configured:

```
bstun protocol-group 1 bsc-local-ack

interface Serial1
 encapsulation frame-relay ietf
 clock rate 125000
 frame-relay map llc2 16

interface Serial4
 no ip address
encapsulation bstun
bstun group 1
bsc secondary
bstun route address C3 interface Serial1 dlci 16 C
bstun route address C2 interface Serial1 dlci 16 8
bstun route address C1 interface Serial1 dlci 16 4
```

BSTUN-over-Frame Relay Configuration with Passthrough Example

The following example configures BSTUN over Frame Relay with Passthrough configured:

```
bstun protocol-group 1 bsc

interface Serial1
 encapsulation frame-relay
 clock rate 125000
 frame-relay map bstun 16
 frame-relay map llc 16

interface Serial4
 no ip address
encapsulation bstun
bstun group 1
bsc secondary
bstun route address C3 interface Serial1 dlci 16
bstun route address C2 interface Serial1 dlci 16
bstun route address C1 interface Serial1 dlci 16
```

Configuring LLC2 and SDLC Parameters

You do not need to configure Logical Link Control, type 2 (LLC2) Protocol because it is already enabled on Token Ring interfaces. This chapter describes how to modify the default settings of LLC2 parameters as needed.

To support the Synchronous Data Link Control (SDLC) protocol, you must configure the router to act as a primary or secondary SDLC station. You also can change default settings on any SDLC parameters. Configuration examples for both LLC2 and SDLC are given at the end of the chapter.

For a complete description of the LLC2 and SDLC commands mentioned in this chapter, refer to the "LLC2 and SDLC Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

The LLC2 and SDLC protocols provide data link layer support for higher-layer network protocols and features such as SDLC Logical Link Control (SDLLC) and RSRB with local acknowledgment. The features that are affected by LLC2 parameter settings are listed in the "The Cisco Implementation of LLC2" section on page 2. The features that require SDLC configuration and use SDLC parameters are listed in the "The Cisco Implementation of SDLC" section on page 2.

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

LLC2 and SDLC package data in frames. LLC2 and SDLC stations require acknowledgments from receiving stations after a set amount of frames have been sent before sending further data. The tasks described in this chapter modify default settings regarding the control field of the data frames. By modifying the control field parameters, you can determine the number of acknowledgments sent for frames received and the level of polling used to determine available stations. In this manner, you can set the amount of resources used for frame checking and optimize the network load.

SDLC is used as the primary SNA link-layer protocol for WAN links. SDLC defines two types of network nodes: primary and secondary. Primary nodes poll secondary nodes in a predetermined order. Secondary nodes then send any outgoing data. When configured as primary and secondary nodes, our routers are established as SDLC stations.

The Cisco Implementation of LLC2

The Cisco LLC2 implementation supports the following features:

- Local acknowledgment for Remote Source-Route Bridging (RSRB)

  This feature is used in our implementation of RSRB as described in the chapter "Configuring Source-Route Bridging."

  Because LANs are now connected through RSRB and WAN backbones, the delays that occur are longer than LLC2 allows for bidirectional communication between hosts. Our local acknowledgment feature addresses the problem of delays, resending data, and loss of user sessions.

- IBM LNM support

  Routers using 4- or 16-Mbps Token Ring interfaces configured for Source-Route Bridging (SRB) support Lan Network Manager (LNM) and provide all IBM bridge program functions. With LNM, a router appears as an IBM source-route bridge, and can manage or monitor any connected Token Ring interface.

  LNM support is described in the chapter "Configuring Source-Route Bridging."

- SDLLC media translation

  The SDLLC feature provides media translation between the serial lines running SDLC and Token Rings running LLC2. SDLLC consolidates the IBM SNA networks running SDLC into a LAN-based, multiprotocol, multimedia backbone network.

  SDLLC is described in the chapter "Configuring IBM Network Media Translation."

- ISO Connection-Mode Network Service (CMNS)

  Cisco's CMNS implementation runs X.25 packets over LLC2 so that X.25 can be extended to Ethernet, Fiber Distributed Data Interface (FDDI), and Token Ring media.

The Cisco Implementation of SDLC

The Cisco SDLC implementation supports the following features:

- Frame Relay Access Support (FRAS)

  With FRAS, a router functions as a Frame Relay Access Device (FRAD) for SDLC, Token Ring, and Ethernet-attached devices over a Frame Relay Boundary Network Node (BNN) link.

  Frame Relay access support is described in the chapter "Configuring SNA Frame Relay Access Support."

- SDLLC media translation

  The SDLLC feature provides media translation between the serial lines running SDLC and Token Rings running LLC2. SDLLC consolidates the IBM SNA networks running SDLC into a LAN-based, multiprotocol, multimedia backbone network.

  SDLLC is described in the chapter "Configuring IBM Network Media Translation."

- SDLC local acknowledgment

  SDLC local acknowledgment is used with SDLC STUN. TCP/IP must be enabled. With local acknowledgment, STUN SDLC connections can be terminated locally at the router, eliminating the need for acknowledgments to be sent across a WAN.

  SDLC local acknowledgment is described in the section "Establish the Frame Encapsulation Method" in the chapter "Configuring STUN and BSTUN."

IBM Network Media Translation

The Cisco IOS software includes the following media translation features that enable network communications across heterogeneous media:

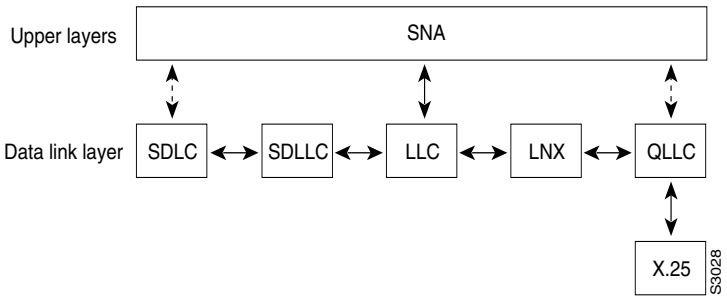
- SDLLC media translation enables a device on a Token Ring to communicate with a device on a serial link.
- QLLC conversion enables an IBM device to communicate with an X.25 network without having to install the X.25 software on local IBM equipment.

SDLLC is Cisco's proprietary software feature that enables a device on a Token Ring to communicate with a device on a serial link by translating between LLC2 and SDLC at the link layer.

SNA uses SDLC and LLC2 as link layer protocols to provide a reliable connection. The translation function between these industry-standard protocols takes place in the proprietary Cisco software.

Figure 161 illustrates how SDLLC provides data link layer support for SNA communication.

Figure 161 SNA Data Link Layer Support



SDLLC Media Translation Features

The SDLLC feature allows a PU 4, PU 2.1, or PU 2 to communicate with a PU 2 SDLC device as follows:

- SDLLC with direct connection—A 37x5 FEP on a Token Ring and the 3x74 cluster controller connected to a serial line are each connected to an interface on the same router configured with SDLLC.
- SDLLC with RSRB—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB using direct encapsulation, RSRB over an FST connection, or RSRB over a TCP connection.
- SDLLC with RSRB and local acknowledgment—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB over a TCP connection that has local acknowledgment enabled.

In all these topologies, each IBM end node (the FEP and cluster controller) has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were communicating over a serial line. That is, the SDLLC software makes translation between the two media transparent to the end nodes.

Virtual Token Ring Concept

Central to Cisco's SDLLC feature is the concept of a virtual Token Ring device residing on a virtual Token Ring. Because the Token Ring device expects the node with which it is communicating also to be on a Token Ring, each SDLLC device on a serial line must be assigned an SDLLC virtual Token Ring address (SDLLC VTRA). Like real Token Ring addresses, SDLLC VTRAs must be unique across the network.

In addition to the SDLLC VTRA, an SDLLC virtual ring number must be assigned to each SDLLC device on a serial line. (The SDLLC virtual ring number differs from the virtual ring group numbers that are used to configure RSRB and multiport bridging.)

As part of its virtual telecommunications access method (VTAM) configuration, the IBM node on the Token Ring has knowledge of the SDLLC VTRA of the serial device with which it communicates. The SDLC VTRA and the SDLLC virtual ring number are a part of the SDLLC configuration for the router's serial interface. When the Token Ring host sends out explorer packets with the SDLLC VTRA as the destination address in the MAC headers, the router configured with that SDLLC VTRA intercepts the frame, fills in the SDLLC virtual ring number address and the bridge number in the RIF, then sends the response back to the Token Ring host. A route is then established between the Token Ring host and the router. After the Cisco IOS software performs the appropriate frame conversion, the system uses this route to forward frames to the serial device.

Resolving Differences in LLC2 and SDLC Frame Size

IBM nodes on Token Ring media normally use frame sizes greater than 1 KB, whereas the IBM nodes on serial lines normally limit frame sizes to 265 or 521 bytes. To reduce traffic on backbone networks and provide better performance, Token Ring nodes should send frames that are as large as possible. As part of the SDLLC configuration on the serial interface, the largest frame size the two media can support

should be selected. The Cisco IOS software can fragment the frames it receives from the Token Ring device before forwarding them to the SDLC device, but it does not assemble the frames it receives from the serial device before forwarding them to the Token Ring device.

Maintaining a Dynamic RIF Cache

SDLLC maintains a dynamic RIF cache and caches the entire RIF; that is, the RIF from the source station to destination station. The cached entry is based on the best path at the time the session begins. SDLLC uses the RIF cache to maintain the LLC2 session between the router and the host FEP. SDLLC does not age these RIF entries. Instead, SDLLC places an entry in the RIF cache for a session when the session begins and flushes the cache when the session terminates. You cannot flush these RIFs because if you flush the RIF entries randomly, the Cisco IOS software cannot maintain the LLC2 session to the host FEP.

Other Considerations

The following are additional facts regarding SDLC and SDLLC:

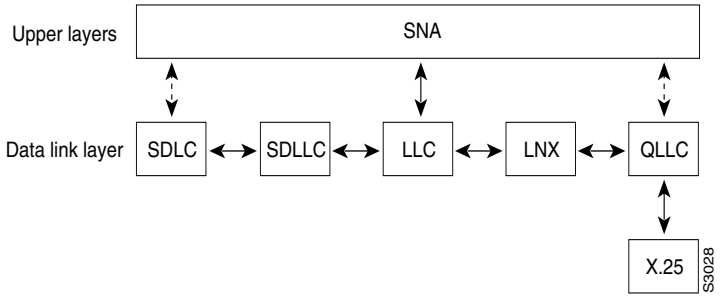
- As part of Cisco's SDLC implementation, only modulus 8 Normal Response Mode (NRM) sessions are maintained for the SDLC session.
- SDLC sessions are always locally acknowledged. LLC2 sessions can be optionally configured for local acknowledgment.
- SDLLC does not apply to SNA subarea networks, such as 37x5 FEP-to 37x5 FEP communication.
- Parameters such as the maximum number of information frames (I-frames) outstanding before acknowledgment, frequency of polls, and response time to poll frames can be modified per interface. If local acknowledgment is not enabled, these parameters are modified on the SDLC interface. If local acknowledgment is enabled, these parameters are modified on the Token Ring interface.
- Local acknowledgment only applies when the remote peer is defined for RSRB using IP encapsulation over a TCP connection. If no local acknowledgment is used, the remote peer can be defined for RSRB using direct encapsulation, RSRB using IP encapsulation over an Fast- Sequenced Transport (FST) connection, or RSRB using IP encapsulation over a TCP connection.

QLLC Conversion

Qualified Logical Link Control (QLLC) is a data link protocol defined by IBM that allows Systems Network Architecture (SNA) data to be transported across X.25 networks. (Although IBM has defined other protocols for transporting SNA traffic over an X.25 network, QLLC is the most widely used.)

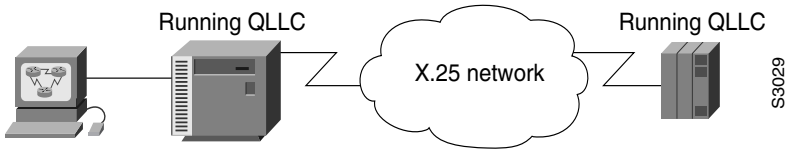
Figure 162 illustrates how QLLC conversion provides data link layer support for SNA communication.

Figure 162 SNA Data Link Layer Support



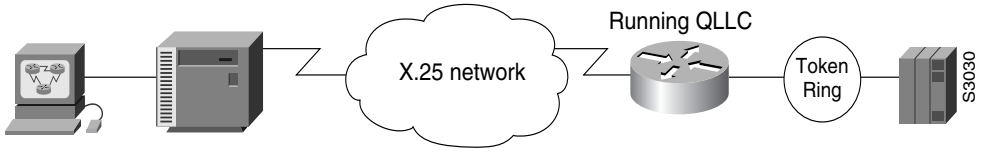
As shown in Figure 163, any devices in the SNA communication path that use X.25, whether end systems or intermediate systems, require a QLLC implementation.

Figure 163 SNA Devices Running QLLC



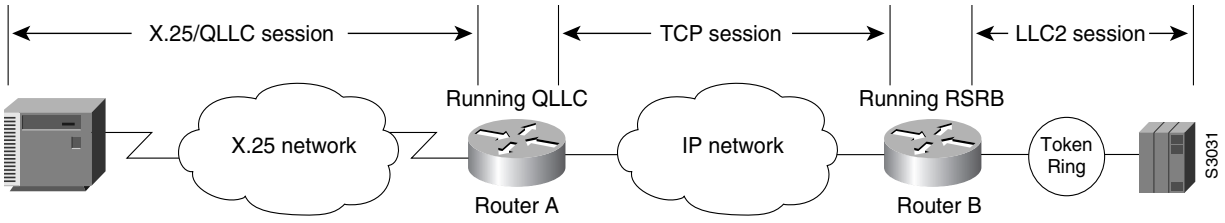
As shown in Figure 164, the QLLC conversion feature eliminates the need to install the X.25 software on local IBM equipment. A device attached locally to a Token Ring network can communicate through a router running the QLLC Conversion feature with a remote device attached to an X.25 network using QLLC. Typically, the locally attached device is an FEP, an AS 400, or a PS/2, and the remote device is a terminal controller or a PS/2. In this case, only the remote device needs an X.25 interface and the FEP can communicate with the terminal controller as if it were directly attached via a Token Ring network.

Figure 164 Router Running QLLC Conversion Feature



More elaborate configurations are possible. The router that implements QLLC conversion need not be on the same Token Ring network as the FEP. As shown in Figure 165, QLLC/LLC2 conversion is possible even when an intermediate IP WAN exists between the router connected to the X.25 network and the router connected to the Token Ring.

Figure 165 QLLC Conversion Running on a Router with an Intermediate IP Network

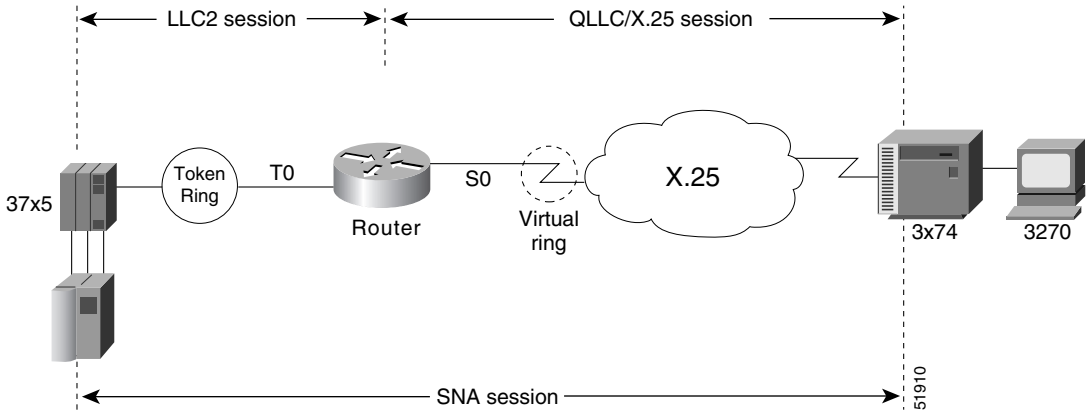


The Cisco Implementation of QLLC Conversion

SNA uses QLLC and X.25 as link layer protocols to provide a reliable connection. QLLC itself processes QLLC control packets. In a Token Ring environment, SNA uses LLC to provide a reliable connection. The LAN-to-X.25 (LNX) software provides a QLLC conversion function to translate between LLC and QLLC.

Figure 166 shows the simplest QLLC conversion topology: a single Token Ring device (for example, a 37x5 FEP) communicates with a single remote X.25 device (in this case a 3x74 cluster controller). In this example, a router connects the Token Ring network to the X.25 network.

Figure 166 QLLC Conversion Between a Single 37x5 and a Single 3x74

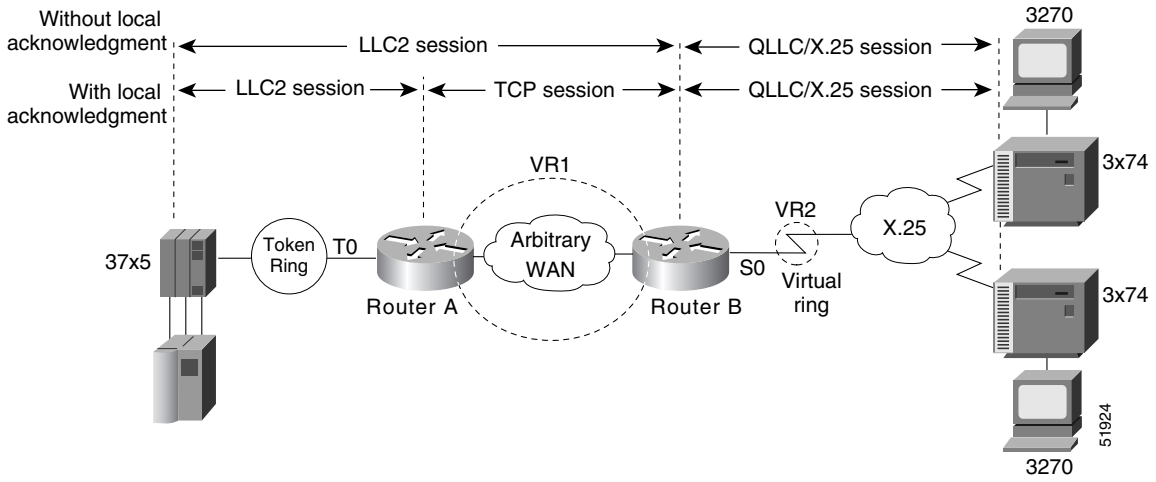


In Figure 166, each IBM end node has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were communicating over an X.25 network. This is accomplished by configuring the router's X.25 interface as a virtual Token Ring, so that the X.25 virtual circuit appears to the Token Ring device (and to the router itself) as if it were a Token Ring to which the remote X.25 device is attached.

Also in this figure, the LLC2 connection extends from the 37x5 FEP across the Token Ring network to the router. The QLLC/X.25 session extends from the router across the X.25 network to the 3x74 cluster controller. Only the SNA session extends across the Token Ring and X.25 networks to provide an end-to-end connection from the 37x5 FEP to the 3x74 cluster controller.

As Figure 167 shows, a router need not directly connect the two IBM end nodes; instead, some type of backbone WAN can connect them. Here, RSRB transports packets between Router A and Router B, while Router B performs all conversion between the LLC2 and X.25 protocols. Only the router attached to the serial line (Router B) needs to be configured for QLLC conversion. Both Router A and Router B are configured for normal RSRB.

Figure 167 QLLC Conversion Between a Single 37x5 and Multiple 3x74s Across an Arbitrary WAN



How communication sessions are established over the communication link varies depending on whether or not LLC2 local acknowledgment has been configured on Router A's Token Ring interface. In both cases, the SNA session extends end-to-end and the QLLC/X.25 session extends from Router B to the 3x74 cluster controller. If LLC2 local acknowledgment has not been configured, the LLC2 session extends from the 37x5 FEP across the Token Ring network and the arbitrary WAN to Router B. In contrast, when LLC2 local acknowledgment has been configured, the LLC2 session extends from the 37x5 FEP Router A, where it is locally terminated. A TCP session is then used across the arbitrary WAN to Router B.

Comparing QLLC Conversion to SDLLC

Although the procedures you use to configure QLLC are similar to those used to configure SDLLC, there are structural and philosophical differences between the point-to-point links that SDLC uses and the multiplexed virtual circuits that X.25 uses.

The most significant structural difference between QLLC conversion and SDLLC is the addressing. To allow a device to use LLC2 to transfer data, both SDLLC and QLLC provide virtual MAC addresses. In SDLLC, the actual MAC address is built by combining the defined virtual MAC (whose last byte is 0x00) with the secondary address used on the SDLC link; in this way, SDLLC supports multidrop. In QLLC conversion, multidrop is meaningless, so the virtual MAC address represents just one session and is defined as part of the X.25 configuration. Because one physical X.25 interface can support many simultaneous connections for many different remote devices, you only need one physical link to the X.25 network. The different connections on different virtual circuits all use the same physical link.

The most significant difference between QLLC conversion and SDLLC is the fact that a typical SDLC/SDLLC operation uses a leased line. In SDLC, dial-up connections are possible, but the maximum data rate is limited. In QLLC, both switched virtual circuits (SVCs) and permanent virtual circuits (PVCs) are available, but the favored use is SVC. While the router maintains a permanent connection to the X.25 network, a remote device can use each SVC for some bounded period of time and then relinquish it for use by another device. Using a PVC is very much like using a leased line.

Table 6 shows how the QLLC commands correspond to the SDLLC commands.

Table 6 QLLC and SDLLC Command Comparison

QLLC Command	Analogous SDLLC Command
`qllc largest-packet`	`sdllc ring-largest-frame, sdllc sdlc-largest-frame`
`qllc partner`	`sdllc partner`
`qllc sap`	`sdllc sap`
`qllc srb, x25 map qllc, x25 pvc qllc`	`sdllc traddr`
`qllc xid`	`sdllc xid`
`source-bridge qllc-local-ack`	`source-bridge sdllc-local-ack`

Other Implementation Considerations

Consider the following when implementing QLLC conversion:

- To use the QLLC conversion feature, a router must have a physical link to an X.25 public data network (PDN). It must also have an SRB/RSRB path to an IBM Front-End Processor (FEP). This link could be a Token Ring or Ethernet interface, or even FDDI, if RSRB is being used.
- QLLC conversion can run on any router with at least one serial interface configured for X.25 communication and at least one other interface configured for SRB or RSRB.
- QLLC conversion security depends upon access control in SRB/RSRB and X.25 and upon exchange identification (XID) validation.

You can configure DLSw+ for QLLC connectivity, which enables the following scenarios:

- Remote LAN-attached devices (physical units) or SDLC-attached devices can access an FEP or an AS/400 over an X.25 network.
- Remote X.25-attached SNA devices can access an FEP or an AS/400 over a Token Ring or over SDLC.

For information on configuring DLSw+ for QLLC conversion, refer to the “Configuring DLSw+” chapter.

You can configure DSPUs for QLLC. For more information on this configuration, refer to the “Configuring DSPU and SNA Service Point Support” chapter.

LLC2 Configuration Task List

Because LLC2 is already enabled on a Token Ring, you do not need to enable it on the router. However, you can enhance LLC2 performance by completing the following tasks:

- Controlling Transmission of I-Frames, page 10
- Establishing the Polling Level, page 12
- Setting Up XID Transmissions, page 13

See the “LLC2 and SDLC Configuration Examples” section on page 21 for examples.

Controlling Transmission of I-Frames

Control the number of information frames (I-frames) and acknowledgments sent on the LLC2 network by completing the tasks described in the following sections:

Setting the Maximum Number of I-Frames Received Before Sending an Acknowledgment

You can reduce overhead on the network by increasing the maximum number of frames the Cisco IOS software can receive at once before it must send the sender an acknowledgment. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 ack-max` *packet-count*	Sets maximum number of I-frames the router can receive before it sends an acknowledgment.

Setting the Maximum Delay for Acknowledgments

You can ensure timely receipt of acknowledgments so that sending data is not delayed. Even if the maximum amount of frames has not been reached, you can set a timer forcing the router to send an acknowledgment and reset the maximum amount counter to 0.

To set the maximum delay time, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 ack-delay-time` *milliseconds*	Sets the I-frame acknowledgment time.

Setting the Maximum Number of I-Frames Sent Before Requiring Acknowledgment

You can set the maximum number of I-frames that the router sends to an LLC2 station before the software requires an acknowledgment from the receiving end. A higher value reduces overhead on the network. Ensure that the receiving LLC2 station can handle the number of frames set by this value.

To set this value, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 local-window` *packet-count*	Sets the maximum number of I-frames the router sends before it requires an acknowledgment.

Setting the Number of Retries Allowed

You can set the number of times the router will re-send a frame when the receiving station does not acknowledge the frame. Once this value is reached, the session is dropped. This value also is used to determine how often the software will retry polling a busy station. Use this command in conjunction with the **llc2 t1-time** command described in the "Setting the Time for Resending I-Frames" section on page 11. Using them together ensures that the sending of frames is monitored at a reasonable level, while limiting the number of unsuccessful repeated tries.

To set the number of retries, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 n2 retry-count`	Establishes the number of times the router will re-send unacknowledged frames or try polling a busy station.

Setting the Time for Resending I-Frames

You can set the amount of time the router waits before resending unacknowledged I-frames. This interval is called the *T1 time*. Use this command in conjunction with setting the number of retries and setting the transit poll-frame timer. Using these commands in conjunction with each other provides a balance of network monitoring and performance.

To set the T1 time, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 t1-time milliseconds`	Controls how long the router waits for an acknowledgment of transmitted I-frames.

Note Ensure that you allow enough time for the round trip between the router and its LLC2-speaking stations. Under heavy network loading conditions, resending I-frames every 3000 ms is appropriate.

Setting the Time for Resending Rejected Frames

You can set the amount of time that the router will wait for an expected frame before sending a reject command (REJ). Typically, when an LLC2 station sends an I-frame, a sequence number is included in the frame. The LLC2 station that receives these frames will expect to receive them in order. If it does not, it can reject a frame and indicate which frame it is expecting to receive instead. If the correct frame is not sent to the software before the reject timer expires, the software sends a REJ to the remote station and disconnects the LLC2 session.

To set the reject timer, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 trej-time milliseconds`	Sets the time the Cisco IOS software waits for a resend of a rejected frame before sending a reject command to the remote station.

Establishing the Polling Level

You can control the amount of polling that occurs on the LLC2 network by completing the tasks described in the following sections:

- Setting the Polling Frequency, page 12
- Setting the Polling Interval, page 12
- Setting the Transmit-Poll-Frame Timer, page 12

Setting the Polling Frequency

You can set the optimum interval of time after which the router sends Receiver Ready messages or frames that tell other LLC2 stations that the router is available. These polls occur during periods of idle time on the network.

To set polling frequency, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 idle-time` *milliseconds*	Controls the polling frequency during idle traffic.

Setting the Polling Interval

The amount of time the router waits until repolling a busy station can also be set. Use this command in conjunction with setting the number of retries. Typically, you do not need to use this command unless an LLC2 station has unusually long busy periods before clearing the busy state. In this case, you should increase the value so that the station does not time out.

To set the polling interval, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 tbusy-time` *milliseconds*	Sets the amount of time the router will wait before repolling a busy station.

Setting the Transmit-Poll-Frame Timer

When the router sends a command that must receive a response, a poll bit is sent in the frame. When the software sends the poll bit, it cannot send any other frame with the poll bit set until the receiver replies to that poll frame with a frame containing a final bit set. When the timer expires, the software assumes that it can send another frame with a poll bit.

Set the transmit-poll-frame timer to reduce problems with receiving stations that are faulty and cannot send the frame with the final bit set by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 tpf-time` *milliseconds*	Sets the amount of time the router waits for a final response to a poll frame before the resending it.

This value should be larger than the T1 time. The T1 time determines how long the software waits for receipt of an acknowledgment before sending the next set of frames. See the "Setting the Time for Resending I-Frames" section on page 11 for more information.

Setting Up XID Transmissions

You can control the number of frames used for identification on the LLC2 network by completing the tasks described in the following sections:

Setting the Frequency of XID Transmissions

XID frames identify LLC2 stations at a higher level than the MAC address and contain information about the configuration of the stations.You can set how often the router sends an XID frame by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 xid-neg-val-time` *milliseconds*	Sets the frequency of XID transmissions.

Caution Do not change the value unless requested by your technical support representative.

Setting the Time for XID Retries

You can set the amount of time the router waits for a reply to the XID frames it sends to remote stations. The value should be larger than the T1 time, which indicates how long the software waits for an acknowledgment before dropping the session.

To set the time for XID retries, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# llc2 xid-retry-time` *milliseconds*	Sets how long the router waits for a reply to the XID frames it sends to remote stations.

Monitoring and Maintaining LLC2 Stations

You can display the configuration of LLC2 stations to determine which LLC2 parameters need adjustment. Use the following command in privileged EXEC mode:

Command	Purpose
`Router# show llc2`	Displays the configuration of LLC2 stations.

SDLC Configuration Task List

The SDLC tasks described in this section configure the router as an SDLC station. (This is in contrast to a router configured for SDLC Transport, where the device is not an SDLC station, but passes SDLC frames between two SDLC stations across a mixed-media, multiprotocol environment.) The first task is required; you accomplish it with the appropriate set of commands for your network needs. The remaining tasks are optional: you can perform them as necessary to enhance SDLC performance.

See the "LLC2 and SDLC Configuration Examples" section on page 21 for examples.

Enabling the Router as a Primary or a Secondary SDLC Station

SDLC defines two types of network nodes: primary and secondary. Primary nodes poll secondary nodes in a predetermined order. Secondaries then send if they have outgoing data. When configured as primary and secondary nodes, our devices are established as SDLC stations.

Depending on your particular network needs, perform the tasks in one of the following sections to enable the router as an SDLC station:

Establishing an SDLC Station for Frame Relay Access Support

You can establish the router to be any of the following:

- Primary SDLC station
- Secondary SDLC station
- Either primary or secondary, depending on the role of the end stations or on XID negotiations
- Primary Node Type 2.1 (NT2.1) node

To establish devices as SDLC stations when you plan to configure Frame Relay access support, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# encapsulation sdlc`[1]	Sets the encapsulation type of the serial interface to SDLC.
Step 2	`Router(config-if)# sdlc role {none | primary | secondary | prim-xid-poll}`	Establishes the role of the interface.

1. For information on the **nrzi-encoding** interface configuration command, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide*.

If the interface does not play a role, the router can be either primary or secondary, depending on the end stations. The SDLC end station must be configured as negotiable or primary NT2.1. When the end stations are configured as physical unit (PU) type 2, you can set the role of the interface to primary or secondary. When the end station is configured as secondary NT2.1, you must set the role of the interface to poll the primary XID.

Note Currently, Frame Relay access support does not support the secondary role.

Establishing an SDLC Station for DLSw+ Support

To establish devices as SDLC stations when you plan to configure our DLSw+ feature, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# sdlc role {none | primary | secondary | prim-xid-poll}`	Sets the encapsulation type of the serial interface to SDLC.
Step 2	`Router(config-if)# sdlc vmac mac-address`	Establishes the role of the interface.
Step 3	`Router(config-if)# sdlc partner mac-address sdlc-address {inbound | outbound}`	Configures a MAC address for the serial interface.
Step 4	`Router(config-if)# sdlc dlsw {sdlc-address | default | partner mac-address [inbound | outbound]}`	Specifies the destination address with which an LLC session is established for the SDLC station.
Step 5		Attaches SDLC addresses to DLSw+.

To configure an SDLC multidrop line downstream, you configure the SDLC role as either **primary** or **prim-xid-poll**. SDLC role **primary** specifies that any PU without the xid-poll parameter in the **sdlc address** command is a PU 2.0 device. SDLC role **prim-xid-poll** specifies that every PU is type 2.1. We recommend that you specify **sdlc role primary** if all SDLC devices are type PU 2.0 or a mix of PU 2.0 and PU 2.1. Use the **sdlc role prim-xid-poll** command if all devices are type PU 2.1.

For additional DLSw+ configuration commands, refer to the “Configuring DLSw+” chapter in this publication.

Establishing an SDLC Station for SDLLC Media Translation

To establish devices as SDLC stations when you plan to configure our SDLLC media translation feature, use the commands in the order listed in the following table. One serial interface can have two or more secondary stations attached to it through a modem sharing device. Each secondary station address must be assigned to the primary station. You must use the following commands in interface configuration mode for the serial interface:

	Command	Purpose
Step 1	`Router(config-if)# encapsulation sdlc-primary`	Establishes a router as the primary SDLC station on the serial line.
Step 2	`Router(config-if)# encapsulation sdlc-secondary`	Establishes other routers as secondary SDLC stations.
Step 3	`Router(config-if)# sdlc address` *hexbyte* [*echo*]	Assigns secondary stations to a primary station.

Use the **show interfaces** command to list the configuration of the SDLC serial lines. Use the **no sdlc address** command to remove a secondary address assignment. Addresses are hexadecimal (base 16).

Enabling SDLC Two-Way Simultaneous Mode

SDLC two-way simultaneous mode allows SDLC link stations to a full-duplex serial line efficiently. With a two-way simultaneous mode, the primary link station can send data to a secondary link station while there is an outstanding poll.

For a primary link station, SDLC two-way simultaneous mode operates in either a multidrop link environment or point-to-point link environment.

In a multidrop link environment, a two-way simultaneous primary station is able to poll a secondary station, receive data from the station, and send data (I-frames) to other secondary stations by using the **sdlc simultaneous half-datamode** command.

In a point-to-point link environment, a two-way simultaneous primary station can send data (I-frames) to a secondary station, although there is an outstanding poll, as long as the window limit is not reached by using the **sdlc simultaneous full-datamode** command.

For a secondary link station, the SDLC two-way simultaneous mode operates only in a point-to-point link environment and allows data (I-frames) to be received after a poll frame has already been received by using the **sdlc simultaneous full-datamode** command.

To enable a two-way simultaneous mode, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# sdlc simultaneous full-datamode`	Enables the primary station in a point-to-point link environment to send data to and receive data from the polled secondary station.
or	Enables the secondary station in a point-to-point link environment to receive data from the primary station after it has already been polled.
`Router(config-if)# sdlc simultaneous half-datamode`	Enables the primary station in a multidrop link environment to send data to other secondary link stations while receiving data from the polled secondary link station.

Determining the Use of Frame Rejects

You can specify that a secondary station does not send frame reject messages, or reject commands indicating frame errors. If you do so, the router drops an SDLC connection if the system receives an error from the secondary station.

To determine handling of frame rejects, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc frmr-disable`	Specifies that this secondary station does not support frame rejects.

To specify that the secondary station does support frame rejects, use the **no sdlc frmr-disable** command.

Setting SDLC Timer and Retry Counts

When an SDLC station sends a frame, it waits for an acknowledgment from the receiver indicating that this frame has been received. You can modify the time the router allows for an acknowledgment before resending the frame. You can also determine the number of times that a software re-sends a frame before terminating the SDLC session. By controlling these values, you can reduce network overhead while continuing to check sending of frames.

Use the SNRM timer only if you want to have a unique timeout period to wait for a reply to a SNRM. To specify a SNRM timer that is different from the T1 response time, set the SDLC SNRM timer using the **sdlc snrm-timer** command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc t1` *milliseconds*	Controls the amount of time the Cisco IOS software waits for a reply. Default value is 3000 ms.
`Router(config-if)# sdlc n2` *retry-count*	Determines the number of times that the Cisco IOS software resends a frame before terminating the SDLC session.
`Router(config-if)# sdlc snrm-timer` *number*	Specifies a SNRM timer that is different from the T1 response time.

Setting SDLC Frame and Window Sizes

You can set the maximum size of an incoming frame and set the maximum number of I-frames (or window size) the router will receive before sending an acknowledgment to the sender. By using higher values, you can reduce network overhead.

To set SDLC frame and window sizes, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# sdlc n1 bit-count`	Sets the maximum size of an incoming frame.
`Router(config-if)# sdlc k window-size`	Sets the local window size of the router.
`Router(config-if)# sdlc poll-limit-value count`	Controls how many times a single secondary station can be polled for input before the next station must be polled.
`Router(config-if)# sdlc address hexbyte [echo] [ack-mode] [xid-poll] [switched] [seconly] [xid-passthru] [passive] [K num]`	Specifies the address used on the SDLC line, and any other unique options on how the address is treated. **Note** The **ack-mode** option supports applications that require local termination of an SDLC connection with address ff. This option is available only if the hexbyte parameter is configured with a value of ff. You should use this option only if you use the SDLC address ff as a regular (not a broadcast) address.

Controlling the Buffer Size

You can control the buffer size on the router. The buffer holds data that is waiting to be sent to a remote SDLC station. This command is particularly useful in the case of the SDLLC media translator, which allows an LLC2-speaking SNA station on a Token Ring to communicate with an SDLC-speaking SNA station on a serial link. The frame sizes and window sizes on Token Rings are often much larger than those acceptable for serial links, and serial links are often slower than Token Rings.

To control backlogs that can occur during periods of high data transfer from the Token Ring to the serial line, use the following command in interface configuration mode on a per-address basis:

Command	Purpose
`Router(config-if)# sdlc holdq address queue-size`	Sets the maximum number of packets held in queue before transmitting.

Controlling Polling of Secondary Stations

You can control the intervals at which the router polls secondary stations, the length of time a primary station can send data to a secondary station, and how often the software polls one secondary station before moving on to the next station.

Keep the following points in mind when using these commands:

- Secondary stations cannot send data until they are polled by a primary station. Increasing the poll-pause timer increases the response time of the secondary stations. Decreasing the timer can flood the serial link with unneeded polls, requiring secondary stations to spend wasted CPU time processing them.
- Increasing the value of the poll limit allows for smoother transactions between a primary station and a single secondary station, but can delay polling of other secondary stations.

To control polling of secondary stations, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# sdlc poll-pause-timer` *milliseconds*	Controls how long the Cisco IOS software pauses between sending each poll frame to secondary stations on a single serial interface.
`Router(config-if)# sdlc poll-limit-value` *count*	Controls how many times a single secondary station can be polled for input before the next station must be polled.

To retrieve default polling values for these operations, use the **no** forms of these commands.

Configuring an SDLC Interface for Half-Duplex Mode

By default, SDLC interfaces operate in full-duplex mode. To configure an SDLC interface for half-duplex mode, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# half-duplex`	Configures an SDLC interface for half-duplex mode.

On an interface that is in half-duplex mode and that has been configured for DCE, you can adjust the delay between the detection of a Request To Send (RTS) signal and the assertion of the Clear To Send (CTS) signal. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# half-duplex timer cts-delay` *value*	Delays the assertion of a CTS.

On an interface that is in half-duplex mode and that has been configured for DTE, you can adjust the time the interface waits for the DCE to assert CTS before dropping an RTS. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# half-duplex timer rts-timeout` *value*	Adjusts the amount of time before interface drops an RTS.

Specifying the XID Value

The exchange of identification (XID) value you define on the router must match that of the IDBLK and IDNUM system generation parameters defined in VTAM on the Token Ring host to which the SDLC device will be communicating. To specify the XID value, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc xid address xid`	Specifies the XID value to be associated with the SDLC station.

Specifying the SAPs

SAPs are used by the CMCC adapter to establish communication with VTAM on the mainframe and to identify Logical Link Control (LLC) sessions on a CMCC's internal adapter. To configure SAPs in SDLC, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc saps address ssap dsap`	Configures SDLC-to-LLC sessions with respect to the SSAP and DSAP on the LLC.

Setting the Largest SDLC I-Frame Size

Generally, the router and the SDLC device with which it communicates should support the same maximum SDLC I-frame size. The larger this value, the more efficient the line usage, thus increasing performance.

After the SDLC device has been configured to send the largest possible I-frame, you must configure the router to support the same maximum I-frame size. The default is 265 bytes. The maximum value the software can support must be less than the value of the LLC2 largest frame value defined when setting the largest LLC2 I-frame size.

To set the largest SDLC I-frame size, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc sdlc-largest-frame address size`	Sets the largest I-frame size that can be sent or received by the designated SDLC station.

Monitoring and Maintaining SDLC Stations

To monitor the configuration of SDLC stations to determine which SDLC parameters need adjustment, use the following command in privileged EXEC mode:

Command	Purpose
`Router# show interfaces serial`	Displays SDLC station configuration information.

You determine the status of end stations by sending an SDLC test frame to a physical unit via its SDLC address and router interface. You can either send out the default information string or a predefined one. You can send a preset number of test frames a continuous stream that can later be halted. The **sdlc test serial** command pre-check for correct interface and SDLC address of the end station. You can view the results of the test frames after the frames have been sent or a SDLC test frame stop has been executed.

To send an SDLC test frame, use the following command in privileged EXEC mode:

Command	Purpose
`Router# sdlc test serial number address [iterations | continuous | stop | string string]`	Sends an SDLC test frame.

Note Only a device configured as primary is allowed to send test frames.

LLC2 and SDLC Configuration Examples

The following sections provide LLC2 and SDLC configuration examples:

LLC2 Configuration Example

You can configure the number of LLC2 frames received before an acknowledgment. For this example, assume that at time 0, two I-frames are received. The maximum amount of three has not been reached, so no acknowledgment for these frames is sent. If a third frame, which would force the router to send an acknowledgment, is not received within 800 ms, an acknowledgment is sent anyway, because the delay timer alarm is activated.

```
interface tokenring 0
 llc2 ack-max 3
 llc2 ack-delay-time 800
```

At this point, because all frames are acknowledged, the counter for the maximum amount of I-frames will be reset to zero.

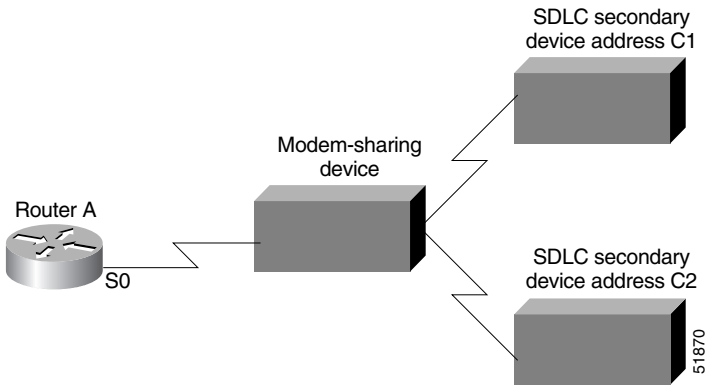
SDLC Two-Way Simultaneous Mode Configuration Example

The following configuration defines serial interface 0 as the primary SDLC station with two SDLC secondary stations, C1 and C2, attached to it through a modem-sharing device. Two-way simultaneous mode is enabled.

```
interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
 sdlc address c2
 sdlc simultaneous half-datamode
```

The network for this configuration is shown in Figure 168.

Figure 168 Two SDLC Secondary Stations Attached to a Single Serial Interface Through a Modem-Sharing Device



SDLC Encapsulation for Frame Relay Access Support Configuration Examples

The following examples describe possible SDLC encapsulation configurations if you plan to configure Frame Relay access support.

The following configuration is appropriate if the SDLC station is a negotiable or primary Node Type 2.1 station:

```
interface serial 2/6
 no ip address
 encapsulation sdlc
 clockrate 9600
 fras map sdlc C1 serial 2/0 frame-relay 32 4 4
 sdlc address C1
```

The following configuration is appropriate if the SDLC station is a secondary Node Type 2.1 station:

```
interface serial 2/6
 no ip address
 encapsulation sdlc
 clockrate 9600
 fras map sdlc C1 serial 2/0 frame-relay 32 4 4
 sdlc role prim-xid-poll
 sdlc address C1
```

The following configuration is appropriate if the SDLC station is a secondary PU 2 station:

```
interface serial 2/6
 no ip address
 encapsulation sdlc
 clockrate 9600
 fras map sdlc C1 serial 2/0 frame-relay 32 4 4
 sdlc role primary
 sdlc address C1
 sdlc xid C1 01700001
```

SDLC Configuration for DLSw+ Example

The following example describes the SDLC configuration with DLSw+ support implemented. In this example, 4000.3745.001 is the MAC address of the host. The router serves as the primary station, while the remote secondary stations, C1, C2, and C3, are reserved for DLSw+ and cannot be used by any other data-link user. The SNRM timer is configured with a value of 2500 ms.

If the **k** parameter is not specified on the **sdlc address** command, the value will be the setting of the **sdlc k** parameter, which is specified as 1; thus C1 and C2 will use **k** value of 1, but the C3 station will have more bandwidth because it has a specified **k** value of 7.

```
interface serial 0
encapsulation sdlc
sdlc role primary
sdlc vmac 4000.3174.0000
sdlc k 1
sdlc address c1
sdlc xid c1 01712345
sdlc partner 4000.3745.0001 c1
sdlc address c2
sdlc xid c2 01767890
sdlc partner 4000.3745.0001 c2


sdlc addr c3 k 7
sdlc xid c3 01754321
sdlc partner 4000.3745.0001 c3
sdlc snrm-timer 2500
sdlc dlsw c1 c2 c3
```

Note If the **no** form of this command is specified, the value of the t1 timer will be used for the SNRM timer.

Half-Duplex Configuration Example

In the following example, an SDLC interface has been configured for half-duplex mode:

```
encapsulation sdlc-primary
 half-duplex
```

SDLC-to-LLC2 FID4 Frame Conversion Examples

The following sample configurations demonstrate SDLC-to-LLC2 conversions for FID4 frames. When you implement these conversion, keep the following considerations in mind:

- If NCP is the primary, the first PU 4 line uses SDLC address 0x01, the second uses 0x02, and so on.

- The SDLC address is used to modify the last byte of the SDLC virtual MAC address (**sdlc vmac**). This modified value is coded in the XCA subarea major node.
- Specify the **echo** option in the **sdlc address** command. With the **echo** option specified, the primary polls with an address in the range 01 to 7E, and the secondary replies with the first bit set to 1. For example, if the primary polls with 04 (0000 0100), the secondary replies with 84 (1000 0100).
- Set **mtu** slightly larger than the maximum packet size used by NCP. Set **sdlc N1** equal to (**mtu** + 2) * 8, which is **mtu**, plus 2 bytes for the SDLC header, times 8 (because N1 is coded in bits, not bytes).
- If the router is providing a clock for the FEP, specify a **clockrate**.
- If the SDLC line has **NRZI=YES**, specify **nrzi-encoding**.
- Ensure that the SDLC- attached FEP is the SDLC primary device, using one of the following methods:
  - Ensure that the SDLC FEP has a higher subarea than the Token Ring-attached FEP (or Token Ring-attached host).
  - Do not configure a secondary SDLCST entry on the GROUP statement for the SDLC line:

```
SDLCPRIM SDLCST GROUP=xxxx
SDLCSEC  SDLCST GROUP=yyyy

GROUP SDLCST=(SDLCPRIM,,)
NAME1 LINE ADDR=nnn
NAME2 PU PUTYPE=4
```

- The SDLC connection requires modulo 8. Ensure that the SDLC group/line and the SDLCST groups are configured with **modulo = 8** and **maxout = 7**.

DLSW Remote Peer Connection Configuration Example

The following sample configurations are for a DLSW remote peer connection using two routers. Two different sample configurations are given for the remote DLSW peer:

- Connected to a CIP-attached router
- Connected to a Token Ring-attached subarea, such as NTRI FEP

Configuration for SDLC-Attached Router

The following configuration statements are for the SDLC-attached router:

```
dlsw local-peer peer-id 10.2.2.2
dlsw remote-peer 0 tcp 10.1.1.1
interface Serial1
 description sdlc configuration PU4/PU4
 mtu 6000
 no ip address
 encapsulation sdlc
 no keepalive
 nrzi-encoding
 clockrate 9600
 sdlc vmac 4000.3745.0000
 sdlc N1 48016
 sdlc address 04 echo
 sdlc partner 4000.1111.0020 04
 sdlc dlsw 4
```

Configuration for Remote DLSW Peer Connected to a CIP-Attached Router

The following configuration statements are for a remote DLSW peer connected to a CIP-attached router:

```
source-bridge ring-group 1111
dlsw local-peer peer-id 10.1.1.1
dlsw remote-peer 0 tcp 10.2.2.2
interface Channel5/0
 csna 0100 20
interface Channel5/2
 lan TokenRing 0
  source-bridge 1 1 1111
  adapter 0 4000.1111.0020
```

Configuration for Remote DLSW Peer Connected to a Token Ring-Attached Subarea

The following configuration statements are for a remote DLSW peer connected to a Token Ring-attached subarea, such as NTRI FEP:

```
source-bridge ring-group 1111
dlsw local-peer peer-id 10.1.1.1
dlsw remote-peer 0 tcp 10.2.2.2
interface token ring 6/0
 ring-speed 16
 source-bridge 2 1 1111
```

DLSW Local-Switching Connection Configuration Example

The following sample configurations are for a DLSW local-switching connection, using one router. Two different sample configurations are given:

- Connection to a CIP-attached router
- Connection to a Token Ring-attached subarea, such as NTRI FEP

Configuration for a Connection to a CIP-Attached Router

The following configuration statements are for a connection to a CIP-attached router:

```
source-bridge ring-group 1111
dlsw local-peer
interface Serial1/0
 description sdlc configuration PU4/PU4
 mtu 6000
 no ip address
 encapsulation sdlc
 no keepalive
 nrzi-encoding
 clockrate 9600
 sdlc vmac 4000.3745.0000
 sdlc N1 48016
 sdlc address 04 echo
 sdlc partner 4000.1111.0020 04
 sdlc dlsw 4
interface Channel5/0
 csna 0100 20
interface Channel5/2
 lan TokenRing 0
  source-bridge 1 1 1111
  adapter 0 4000.1111.0020
```

Configuration for a Connection to a Token Ring-Attached Subarea

The following configuration statements are for a connection to a Token Ring-attached subarea, such as NTRI FEP:

```
source-bridge ring-group 1111
dlsw local-peer
interface Serial1/0
 description sdlc configuration PU4/PU4
 mtu 6000
 no ip address
 encapsulation sdlc
 no keepalive
 nrzi-encoding
 clockrate 9600
 sdlc vmac 4000.3745.0000
 sdlc N1 48016
 sdlc address 04 echo
 sdlc partner 4000.1111.0020 04
 sdlc dlsw 4
interface token ring 6/0
 ring-speed 16
 source-bridge 2 1 1111
```

SDLC FEP Configuration

The following configuration statements are for the SDLC FEP:

```
00084 ************************************************************************"
00085 SDLCPRIM SDLCST GROUP=INNPRIM,         SDLC STATEMENTS FOR INN         *
00086                 MAXOUT=7,                                               *
00087                 MODE=PRIMARY,                                           *
00088                 PASSLIM=254,                                            *
00089                 RETRIES=(5,2,5),                                        *
00090                 SERVLIM=4
00091 SDLCSEC  SDLCST GROUP=INNSEC,          SDLC STATEMENTS FOR INN         *
00092                 MAXOUT=7,                                               *
00093                 MODE=SECONDARY,                                         *
00094                 PASSLIM=254,                                            *
00095                 RETRIES=(5,2,5)
00286 ************************************************************************"
00287 *                                                                      *"
00288 *           GROUP MACROS FOR INN CONNECTIONS                           *"
00289 *                                                                      *"
00290 ************************************************************************"
00291 GRPINN   GROUP ACTIVTO=60,             SEC WAIT FOR PRIM               *
00292                 ANS=CONT,                                               *
00293                 CLOCKNG=EXT,                                            *
00294                 DATRATE=HIGH,                                           *
00295                 DIAL=NO,                                                *
00296                 DUPLEX=FULL,                                            *
00297                 IRETRY=NO,                                              *
00298                 ISTATUS=ACTIVE,                                         *
00299                 LNCTL=SDLC,                                             *
00300                 MAXOUT=7,                                               *
00301                 MAXPU=1,                                                *
00302                 MONLINK=YES,                                            *
00303                 NEWSYNC=NO,                                             *
00304                 NRZI=NO,                                                *
00305                 PASSLIM=254,                                            *
00306                 PAUSE=0.2,                                              *
00307                 REPLYTO=1,                                              *
00308                 RETRIES=(3,1,3),                                        *
```

```
00309                SDLCST=(SDLCPRIM,SDLCSEC),                              *
00310                SERVLIM=255,                                            *
00311                TGN=2,                                                  *
00312                TRANSFR=27,                                             *
00313                TYPE=NCP
00314 *"
00315 ERNLN012 LINE ADDRESS=012,ISTATUS=ACTIVE
00316 ERNPU012 PU PUTYPE=4
00317 *"
```

Token Ring FEP Subarea Configuration

The following configuration statements are for the Token Ring FEP subarea:

```
*********************************************************               06260099
* SDLCST STATEMENT FOR SDLC CONNECTED NCP-NCP LINKS    *               06270099
*********************************************************               06280099
N46DPRIS SDLCST GROUP=N46DPRIG,        *                               X06290099
               MAXOUT=7,               * FRAMES RECIEVED BEFORE RESPONSX06300099
               MODE=PRIMARY,           * PRIMARY MODE                  X06310099
               PASSLIM=254,            * MAXIMUM # OF PIUS SENT TO PU  X06320099
               RETRIES=(3,2,30),       * RETRIES                       X06330099
               SERVLIM=4               * REGULAR / SPECIAL SCANS        06340099
N46DSECS SDLCST GROUP=N46DSECG,                                         X06350099
               MAXOUT=7,                                                X06360099
               MODE=SECONDARY,                                          X06370099
               PASSLIM=254,                                             X06380099
               RETRIES=3                                                 06390099
************************************************************************ 46680099
*        TOKEN RING PHYSICAL DEFINTIONS                                * 46690099
************************************************************************ 46700099
N46DPTR1 GROUP ECLTYPE=(PHYSICAL,SUBAREA),                              X46710099
               NPACOLL=YES                                               46720099
N46LYA   LINE  ADDRESS=(1088,FULL),      TIC ADDRESS                    X46730099
               ISTATUS=ACTIVE,                                          X46743099
               OWNER=H53,                                               X46750099
               PORTADD=1,                                               X46760099
               MAXTSL=1108,                                             X46770099
               RCVBUFC=4095,          MAX FROM RING TO NCP              X46780099
               LOCADD=400000001C46    3745 ADDRESS ON RING               46790099
N46PYA   PU    ANS=CONT                                                  46800099
N46UYA   LU    ISTATUS=INACTIVE     DUMMY LU                             46810099
*              STATOPT=OMIT                                              46820099
************************************************************************ 46829999
*        TOKEN RING LOGICAL DEFINITIONS - SUBAREA LINKS                * 46830099
************************************************************************ 46830199
N46DLTR1 GROUP ECLTYPE=(LOGICAL,SUBAREA),  * LOGICAL SUBAREA GROUP *   X46830299
               ISTATUS=INACTIVE,                                        X46830399
               NPACOLL=YES,                                             X46830499
               OWNER=H53,                                               X46830599
               PHYSRSC=N46PYA                                            46830699
N46LXA47 LINE  SDLCST=(N46DPRIS,N46DSECS),ISTATUS=ACTIVE                 46830799
N46PXA47 PU    ADDR=04400037450004                                       46830999
```

VTAM XCA Subarea Major Node

The following configuration statements are for the VTAM XCA subarea major node:

```
00001          VBUILD TYPE=XCA
00002 SUBAPRT  PORT  ADAPNO=0,                                           *
00003                CUADDR=100,                                         *
00004                MEDIUM=RING,                                        *
```

```
00005                SAPADDR=4,                                          *
00006                TIMER=30
00007 SUBAGRP  GROUP DIAL=NO
00008 SUBALN   LINE  USER=SNA
00009 SUBAPU   PU    MACADDR=4000374500004,                              *
00010                PUTYPE=4,                                           *
00011                SAPADDR=4,                                          *
00012                SUBAREA=63,                                         *
00013                TGN=2
```

Configuring IBM Network Media Translation

This chapter describes how to configure the Cisco IOS software for IBM network media translation with Synchronous Logical Data Link Control (SDLLC) or Qualified Logical Link Control (QLLC). For a complete description of the SDLLC and QLLC commands in this chapter, refer to the "IBM Network Media Translation Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

IBM Network Media Translation Overview

The Cisco IOS software includes the following media translation features that enable network communications across heterogeneous media:

- SDLLC media translation enables a device on a Token Ring to communicate with a device on a serial link.
- QLLC conversion enables an IBM device to communicate with an X.25 network without having to install the X.25 software on local IBM equipment.

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

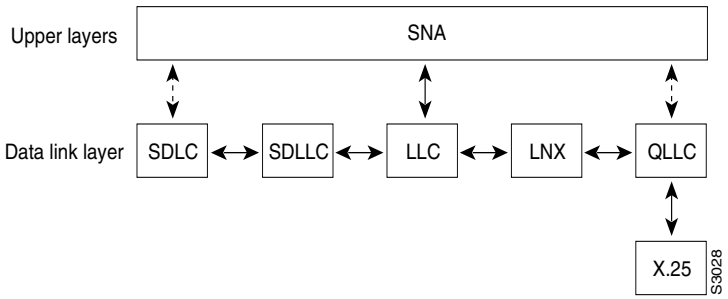
SDLLC is a Cisco Systems proprietary software feature that enables a device on a Token Ring to communicate with a device on a serial link by translating between LLC2 and SDLC at the link layer.

SNA uses SDLC and LLC2 as link layer protocols to provide a reliable connection. The translation function between these industry-standard protocols takes place in the proprietary Cisco software.

This section contains a brief overview of IBM Network Media Translation which is described in the following topics:

Figure 169 illustrates how SDLLC provides data link layer support for SNA communication.

Figure 169 SNA Data Link Layer Support



SDLLC Media Translation Features

The SDLLC feature allows a PU 4, PU 2.1, or PU 2 to communicate with a PU 2 SDLC device as follows:

- SDLLC with direct connection—A 37x5 FEP on a Token Ring and the 3x74 cluster controller connected to a serial line are each connected to an interface on the same router configured with SDLLC.
- SDLLC with RSRB—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB using direct encapsulation, RSRB over an FST connection, or RSRB over a TCP connection.
- SDLLC with RSRB and local acknowledgment—A 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line are connected to different routers. Only the device to which the 3x74 is connected is configured with SDLLC. The routers communicate via RSRB over a TCP connection that has local acknowledgment enabled.

In all these topologies, each IBM end node (the FEP and cluster controller) has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were communicating over a serial line. That is, the SDLLC software makes translation between the two media transparent to the end nodes.

Virtual Token Ring Concept

Central to Cisco's SDLLC feature is the concept of a virtual Token Ring device residing on a virtual Token Ring. Because the Token Ring device expects the node with which it is communicating also to be on a Token Ring, each SDLLC device on a serial line must be assigned an SDLLC virtual Token Ring address (SDLLC VTRA). Like real Token Ring addresses, SDLLC VTRAs must be unique across the network.

In addition to the SDLLC VTRA, an SDLLC virtual ring number must be assigned to each SDLLC device on a serial line. (The SDLLC virtual ring number differs from the virtual ring group numbers that are used to configure RSRB and multiport bridging.)

As part of its virtual telecommunications access method (VTAM) configuration, the IBM node on the Token Ring has knowledge of the SDLLC VTRA of the serial device with which it communicates. The SDLC VTRA and the SDLLC virtual ring number are a part of the SDLLC configuration for the router's serial interface. When the Token Ring host sends out explorer packets with the SDLLC VTRA as the destination address in the MAC headers, the router configured with that SDLLC VTRA intercepts the frame, fills in the SDLLC virtual ring number address and the bridge number in the RIF, then sends the response back to the Token Ring host. A route is then established between the Token Ring host and the router. After the Cisco IOS software performs the appropriate frame conversion, the system uses this route to forward frames to the serial device.

Resolving Differences in LLC2 and SDLC Frame Size

IBM nodes on Token Ring media normally use frame sizes greater than 1 KB, whereas the IBM nodes on serial lines normally limit frame sizes to 265 or 521 bytes. To reduce traffic on backbone networks and provide better performance, Token Ring nodes should send frames that are as large as possible. As part of the SDLLC configuration on the serial interface, the largest frame size the two media can support should be selected. The Cisco IOS software can fragment the frames it receives from the Token Ring device before forwarding them to the SDLC device, but it does not assemble the frames it receives from the serial device before forwarding them to the Token Ring device.

Maintaining a Dynamic RIF Cache

SDLLC maintains a dynamic RIF cache and caches the entire RIF; that is, the RIF from the source station to destination station. The cached entry is based on the best path at the time the session begins. SDLLC uses the RIF cache to maintain the LLC2 session between the router and the host FEP. SDLLC does not age these RIF entries. Instead, SDLLC places an entry in the RIF cache for a session when the session begins and flushes the cache when the session terminates. You cannot flush these RIFs because if you flush the RIF entries randomly, the Cisco IOS software cannot maintain the LLC2 session to the host FEP.

Other Considerations

The following are additional facts regarding SDLC and SDLLC:

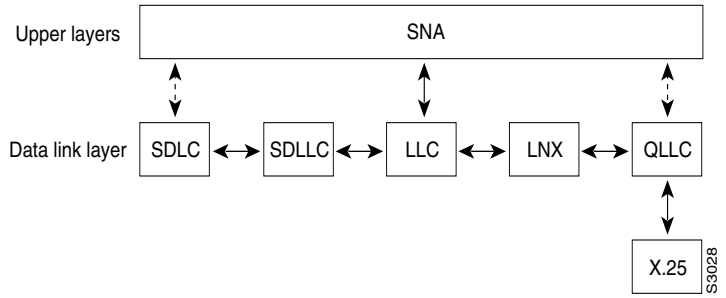
- As part of Cisco's SDLC implementation, only modulus 8 Normal Response Mode (NRM) sessions are maintained for the SDLC session.
- SDLC sessions are always locally acknowledged. LLC2 sessions can be optionally configured for local acknowledgment.
- SDLLC does not apply to SNA subarea networks, such as 37x5 FEP-to 37x5 FEP communication.

- Parameters such as the maximum number of information frames (I-frames) outstanding before acknowledgment, frequency of polls, and response time to poll frames can be modified per interface. If local acknowledgment is not enabled, these parameters are modified on the SDLC interface. If local acknowledgment is enabled, these parameters are modified on the Token Ring interface.
- Local acknowledgment only applies when the remote peer is defined for RSRB using IP encapsulation over a TCP connection. If no local acknowledgment is used, the remote peer can be defined for RSRB using direct encapsulation, RSRB using IP encapsulation over an FST connection, or RSRB using IP encapsulation over a TCP connection.

QLLC Conversion

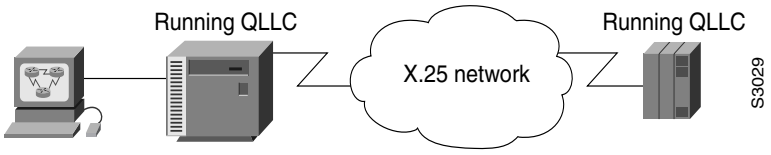
Qualified Logical Link Control (QLLC) is a data link protocol defined by IBM that allows SNA data to be transported across X.25 networks. (Although IBM has defined other protocols for transporting SNA traffic over an X.25 network, QLLC is the most widely used.) Figure 170 illustrates how QLLC conversion provides data link layer support for SNA communication.

Figure 170 SNA Data Link Layer Support



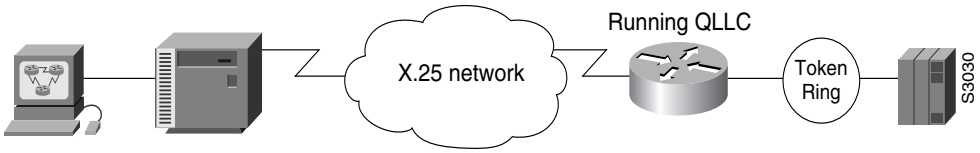
As shown in Figure 171, any devices in the SNA communication path that use X.25, whether end systems or intermediate systems, require a QLLC implementation.

Figure 171 SNA Devices Running QLLC



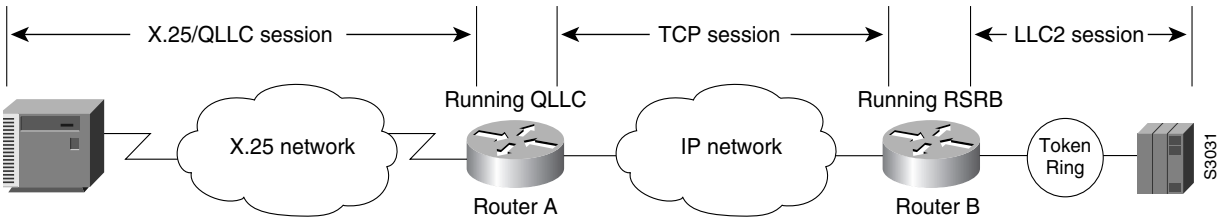
As shown in Figure 172, the QLLC conversion feature eliminates the need to install the X.25 software on local IBM equipment. A device attached locally to a Token Ring network can communicate through a router running the QLLC Conversion feature with a remote device attached to an X.25 network using QLLC. Typically, the locally attached device is an FEP, an AS 400, or a PS/2, and the remote device is a terminal controller or a PS/2. In this case, only the remote device needs an X.25 interface and the FEP can communicate with the terminal controller as if it were directly attached via a Token Ring network.

Figure 172 Router Running QLLC Conversion Feature



More elaborate configurations are possible. The router that implements QLLC conversion need not be on the same Token Ring network as the FEP. As shown in Figure 173, QLLC/LLC2 conversion is possible even when an intermediate IP WAN exists between the router connected to the X.25 network and the router connected to the Token Ring.

Figure 173 QLLC Conversion Running on a Router with an Intermediate IP Network

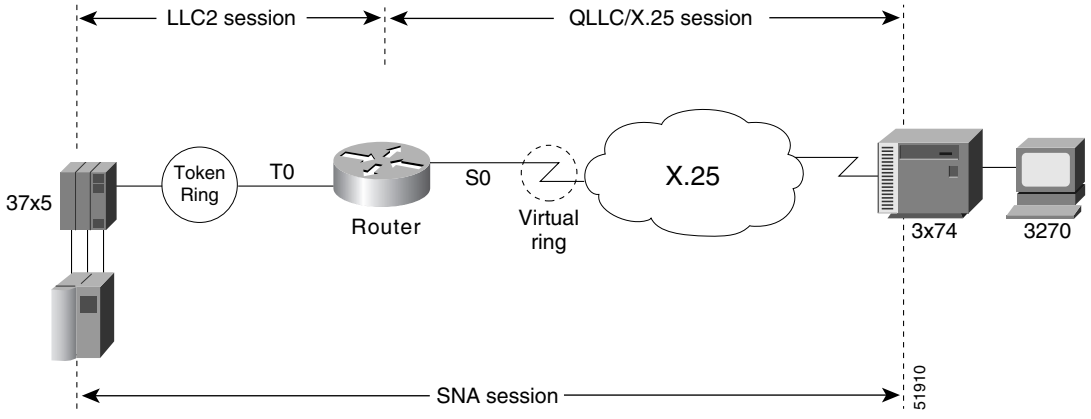


The Cisco Implementation of QLLC Conversion

SNA uses QLLC and X.25 as link layer protocols to provide a reliable connection. QLLC itself processes QLLC control packets. In a Token Ring environment, SNA uses LLC to provide a reliable connection. The LAN-to-X.25 (LNX) software provides a QLLC conversion function to translate between LLC and QLLC.

Figure 174 shows the simplest QLLC conversion topology: a single Token Ring device (for example, a 37x5 FEP) communicates with a single remote X.25 device (in this case a 3x74 cluster controller). In this example, a router connects the Token Ring network to the X.25 network.

Figure 174 QLLC Conversion Between a Single 37x5 and a Single 3x74



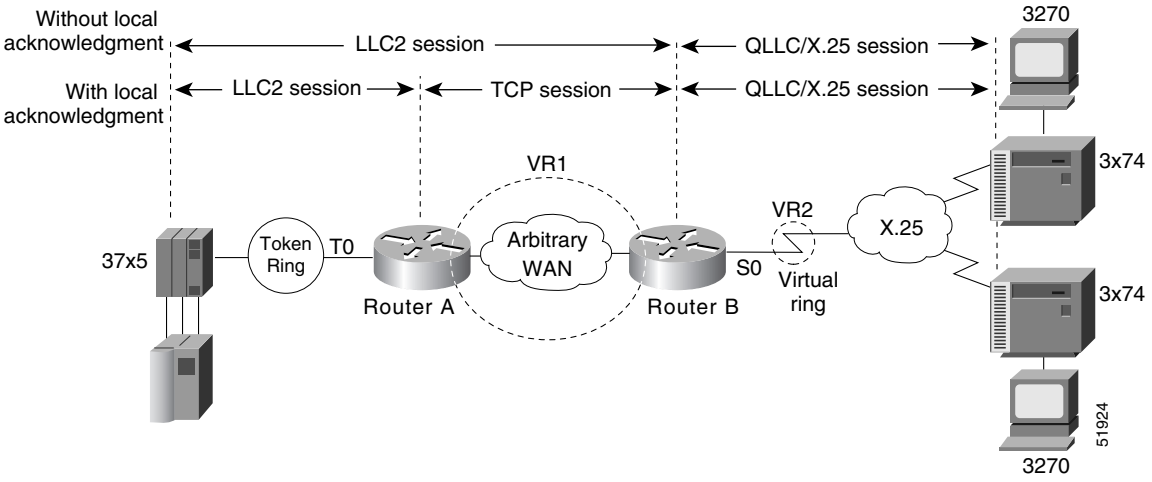
In Figure 174, each IBM end node has no indication that its counterpart is connected to a different medium running a different protocol. The 37x5 FEP responds as if the 3x74 cluster controller were communicating over a Token Ring, whereas the 3x74 responds as though the 37x5 FEP were

communicating over an X.25 network. This is accomplished by configuring the router's X.25 interface as a virtual Token Ring, so that the X.25 virtual circuit appears to the Token Ring device (and to the router itself) as if it were a Token Ring to which the remote X.25 device is attached.

Also in this figure, the LLC2 connection extends from the 37x5 FEP across the Token Ring network to the router. The QLLC/X.25 session extends from the router across the X.25 network to the 3x74 cluster controller. Only the SNA session extends across the Token Ring and X.25 networks to provide an end-to-end connection from the 37x5 FEP to the 3x74 cluster controller.

As Figure 175 shows, a router need not directly connect the two IBM end nodes; instead, some type of backbone WAN can connect them. Here, RSRB transports packets between Router A and Router B, while Router B performs all conversion between the LLC2 and X.25 protocols. Only the router attached to the serial line (Router B) needs to be configured for QLLC conversion. Both Router A and Router B are configured for normal RSRB.

Figure 175 QLLC Conversion Between a Single 37x5 and Multiple 3x74s Across an Arbitrary WAN



How communication sessions are established over the communication link varies depending on whether or not LLC2 local acknowledgment has been configured on Router A's Token Ring interface. In both cases, the SNA session extends end-to-end and the QLLC/X.25 session extends from Router B to the 3x74 cluster controller. If LLC2 local acknowledgment has not been configured, the LLC2 session extends from the 37x5 FEP across the Token Ring network and the arbitrary WAN to Router B. In contrast, when LLC2 local acknowledgment has been configured, the LLC2 session extends from the 37x5 FEP Router A, where it is locally terminated. A TCP session is then used across the arbitrary WAN to Router B.

Comparing QLLC Conversion to SDLLC

Although the procedures you use to configure QLLC are similar to those used to configure SDLLC, there are structural and philosophical differences between the point-to-point links that SDLC uses and the multiplexed virtual circuits that X.25 uses.

The most significant structural difference between QLLC conversion and SDLLC is the addressing. To allow a device to use LLC2 to transfer data, both SDLLC and QLLC provide virtual MAC addresses. In SDLLC, the actual MAC address is built by combining the defined virtual MAC (whose last byte is 0x00) with the secondary address used on the SDLC link; in this way, SDLLC supports multidrop. In QLLC conversion, multidrop is meaningless, so the virtual MAC address represents just one session and is

defined as part of the X.25 configuration. Because one physical X.25 interface can support many simultaneous connections for many different remote devices, you only need one physical link to the X.25 network. The different connections on different virtual circuits all use the same physical link.

The most significant difference between QLLC conversion and SDLLC is the fact that a typical SDLC/SDLLC operation uses a leased line. In SDLC, dial-up connections are possible, but the maximum data rate is limited. In QLLC, both switched virtual circuits (SVCs) and permanent virtual circuits (PVCs) are available, but the favored use is SVC. While the router maintains a permanent connection to the X.25 network, a remote device can use each SVC for some bounded period of time and then relinquish it for use by another device. Using a PVC is very much like using a leased line.

Table 7 shows how the QLLC commands correspond to the SDLLC commands.

Table 7 QLLC and SDLLC Command Comparison

QLLC Command	Analogous SDLLC Command
`qllc largest-packet`	`sdllc ring-largest-frame, sdllc sdlc-largest-frame`
`qllc partner`	`sdllc partner`
`qllc sap`	`sdllc sap`
`qllc srb, x25 map qllc, x25 pvc qllc`	`sdllc traddr`
`qllc xid`	`sdllc xid`
`source-bridge qllc-local-ack`	`source-bridge sdllc-local-ack`

Other Implementation Considerations

Consider the following when implementing QLLC conversion:

- To use the QLLC conversion feature, a router must have a physical link to an X.25 public data network (PDN). It must also have an SRB/RSRB path to an IBM FEP. This link could be a Token Ring or Ethernet interface, or even FDDI, if RSRB is being used.
- QLLC conversion can run on any router with at least one serial interface configured for X.25 communication and at least one other interface configured for SRB or RSRB.
- QLLC conversion security depends upon access control in SRB/RSRB and X.25 and upon XID validation.

You can configure DLSw+ for QLLC connectivity, which enables the following scenarios:

- Remote LAN-attached devices (physical units) or SDLC-attached devices can access an FEP or an AS/400 over an X.25 network.
- Remote X.25-attached SNA devices can access an FEP or an AS/400 over a Token Ring or over SDLC.

For information on configuring DLSw+ for QLLC conversion, refer to the “Configuring DLSw+” chapter.

You can configure DSPUs for QLLC. For more information on this configuration, refer to the “Configuring DSPU and SNA Service Point” chapter.

SDLLC Configuration Task List

To configure SDLLC, perform the tasks in the following sections:

- Configuring SDLLC with Direct Connection, page 9
- Configuring SDLLC with RSRB, page 11
- Configuring SDLLC with RSRB and Local Acknowledgment, page 12
- Configuring SDLLC with Ethernet and Translational Bridging, page 13
- Customizing SDLLC Media Translation, page 14

Note Because data-link switching plus (DLSw+) contains its own media conversion, SDLLC is not required when using DLSw+.

For more information on configuring SDLLC and QLLC, see the following sections:

- QLLC Conversion Configuration Task List, page 16
- SDLLC Configuration Examples, page 20
- QLLC Conversion Configuration Examples, page 27

Configuring SDLLC with Direct Connection

In the SDLLC configuration with direct connection, a 37x5 front-end processor (FEP) on a Token Ring and a 3x74 cluster controller connected to a serial line are each connected to an interface on the same router configured with SDLLC. In this configuration, the Logical Link Control, type 2 (LLC2) session extends from the 37x5 FEP across the Token Ring to the router. The SDLLC session extends from the router across the serial line to the 3x74 cluster controller. The Systems Network Architecture (SNA) session extends across the Token Ring and the serial line to provide an end-to-end connection. The router is configured with source-route bridging (SRB).

To configure SDLLC with direct connection, you must perform the tasks in the following sections:

- Enabling SDLLC Media Translation, page 10
- Associating a SAP Value, page 10
- Specifying the XID Value, page 10
- Initiating a Connection to the Token Ring Host, page 10

For an example of how to configure SDLLC with direct connection, see the "SDLLC with Direct Connection Example" section on page 21.

Enabling SDLLC Media Translation

The interfaces you will configure for SDLLC media translation are the serial interfaces that connect to the serial lines linking the remote Synchronous Data Link Control (SDLC) devices. To configure them, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdllc traddr xxxx.xxxx.xx00 lr bn tr`	Enables SDLLC media translation on a serial interface.

Associating a SAP Value

You can associate a Service Access Point (SAP) value by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdllc sap sdlc-address ssap dsap`	Associates a SAP value.

Specifying the XID Value

The XID value you define in the Cisco IOS software must match that of the IDBLK and IDNUM system generation parameters defined in VTAM of the Token Ring host to which the SDLC device will be communicating. To define XID, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdllc xid address xxxxxxxx`	Specifies the XID value appropriate for the SDLC station to match VTAM values.

Initiating a Connection to the Token Ring Host

The Token Ring host is always kept in a state ready to accept a connection from the remote serial device. The remote serial device is responsible for initiating connections. The advantage of this scheme is that the serial device can communicate with the Token Ring host whenever it chooses without requiring personnel to be on the host site.

The Cisco IOS software actually initiates the connection on behalf of the serial device. To initiate connections, both the Media Access Control (MAC) address of the Token Ring host and the SDLC line address are required. You must configure the Cisco IOS software to define the Token Ring host as the partner of the serial device. To do so, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdllc partner mac-address sdlc-address`	Enables connections for SDLLC.

Configuring SDLLC with RSRB

A router need not directly connect the two IBM end nodes: a 37x5 FEP on a Token Ring and a 3x74 cluster controller connected to a serial line can be connected to different routers. However, the router to which the 3x74 is connected must be configured with SDLLC. They communicate via remote source-route bridging (RSRB) using direct encapsulation, RSRB over an FST connection, or RSRB over a TCP connection. RSRB transports packets between Router A and Router B, while Router B performs all conversion between the LLC2 and SDLC protocols by means of the SDLLC software.

To configure the router for SDLLC with RSRB, you must perform all the tasks in the "Configuring SDLLC with Direct Connection" section on page 9. In addition, you must perform one of the sets of tasks in the following sections:

- Configuring RSRB Using Direct Encapsulation, page 11
- Configuring RSRB over an FST Connection, page 12
- Configuring RSRB over a TCP Connection, page 12

For more information about configuring RSRB, see the chapter "Configuring Source-Route Bridging" in this publication and "Source-Route Bridging Commands" in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Note When you configure RSRB, you must include a **source-bridge remote peer** command on the router connected to the serial line and another **source-bridge remote peer** command on the one connected to the Token Ring. If you have more than one serial line connected to the same router, then you will have a **source-bridge remote peer** command for each interface in its configuration that will be using SDLLC with RSRB.

For an example of how to configure SDLLC with RSRB, see the "SDLLC with RSRB (Multiple 3x74s) Example" section on page 24.

Configuring RSRB Using Direct Encapsulation

To configure SDLLC with RSRB using direct encapsulation, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group interface interface-name [mac-address]`	Defines a remote peer.

Configuring RSRB over an FST Connection

To configure SDLLC with RSRB over an FST connection, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
Step 2	`Router(config)# source-bridge fst-peername local-interface-address`	For FST connection only, sets up an FST peer name.
Step 3	`Router(config)# source-bridge remote-peer ring-group fst ip-address`	Defines a remote peer.

Configuring RSRB over a TCP Connection

To configure SDLLC with RSRB over a TCP connection, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group tcp ip-address`	Defines a remote peer.

Configuring SDLLC with RSRB and Local Acknowledgment

RSRB can be configured for only local acknowledgment with RSRB using IP encapsulation over a TCP connection. Configuring SDLLC local acknowledgment can reduce time-outs and keepalive traffic on the connection.

If LLC2 local acknowledgment is configured, it must be configured on the serial interface of the router on the 3x74 cluster controller side of the connection and on the Token Ring interface of the router on the 37x5 FEP side of the connection. Whether or not local acknowledgment is configured, the SNA session extends end-to-end and the SDLC session extends from the router configured with the serial interface to the 3x74 cluster controller. However, the LLC2 session extends from the 37x5 FEP to the router with the Token Ring interface configured. The LLC2 session is locally terminated at that router. A TCP session is then established across the WAN to a router on the 3x74 side of the connection.

To configure the Cisco IOS software for SDLLC with RSRB and local acknowledgment, you must perform all the tasks in the "Configuring SDLLC with Direct Connection" section on page 9. In addition, you must use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group tcp ip-address local-ack`	Defines a remote peer with the local acknowledgment feature.
Step 3	`Router(config)# source-bridge sdllc-local-ack`	Enables local acknowledgment for connections involving SDLLC media translation.

Local acknowledgment is not supported when the LLC2 device is attached to an Ethernet rather than to a Token Ring.

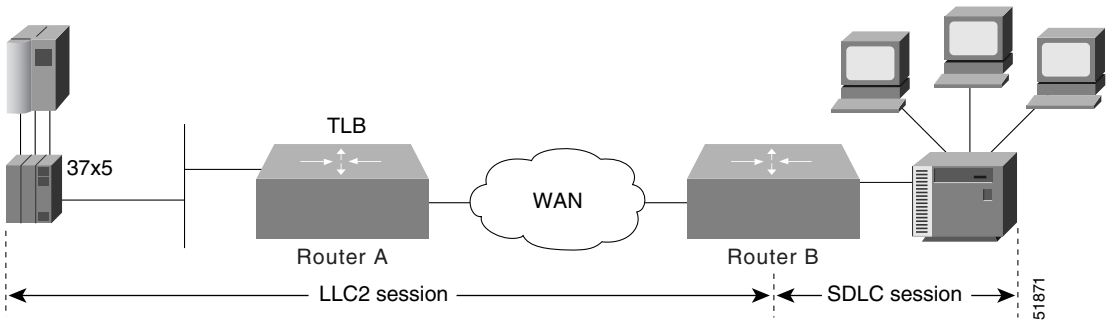
For an example of how to configure SDLLC with RSRB and local acknowledgment, see the "SDLLC with RSRB and Local Acknowledgment Example" section on page 25.

For more information about configuring RSRB and local acknowledgment, see the chapter "Configuring Source-Route Bridging" in this manual and "Source-Route Bridging Commands" in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

Configuring SDLLC with Ethernet and Translational Bridging

SDLLC support over Ethernet combines translational bridging with Ethernet support of 37x5 FEP connections. Figure 176 shows SDLLC with Ethernet and translational bridging. The 3x75 FEP is attached to Router A through Ethernet. The same router is configured for translational bridging, which translates Ethernet packets into Token Ring packets and passes them across the WAN to Router B connected to the 3x74 cluster controller via a serial line. The LLC2 session terminates at Router B connected to the 3x74 cluster controller. In addition, Router B maintains an SDLC session from itself to the cluster controller.

Figure 176 SDLLC with Ethernet and Translational Bridging



Customizing SDLLC Media Translation

To increase performance on connections involving SDLLC media translation, perform the tasks in the following sections:

Setting the Largest LLC2 I-Frame Size

Generally, the router and the LLC2 device with which it communicates should support the same maximum SDLC I-frame size. The larger this value, the better the line is used, thus increasing performance.

Faster screen updates to 3278-style terminals often result by configuring the Token Ring FEP to send as large an I-frame as possible and then allowing the Cisco IOS software to segment the frame into multiple SDLC I-frames.

After the Token Ring FEP has been configured to send the largest possible I-frame, it is best to configure the software to support the same maximum I-frame size. The default is 516 bytes. The maximum value the software can support is 8144 bytes.

To set the largest LLC2 I-frame size, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **`sdllc ring-largest-frame`** *`value`*	Specifies the largest I-frame size that can be sent or received by the designated LLC2 primary station.

Setting the Largest SDLC I-Frame Size

Generally, the router and the SDLC device with which it communicates should support the same maximum SDLC I-frame size. The larger this value, the better the line is utilized, thus increasing performance.

After the SDLC device has been configured to send the largest possible I-frame, you must configure the Cisco IOS software to support the same maximum I-frame size. The default is 265 bytes. The maximum value the software can support must be less than the value of the LLC2 largest frame value defined when setting the largest LLC2 I-frame size.

To set the largest SDLC I-frame size, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **`sdllc sdlc-largest-frame`** *`address value`*	Sets the largest I-frame size that can be sent or received by the designated SDLC station.

Increasing the SDLC Line Speed

You can increase the data transfer rate by increasing the SDLC line speed on the serial interface. If possible, increase the link speed of the 3x74 to 19.2 kbps on older units, or to 64 kbps on new units.

To increase the SDLC line speed, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# clock rate` *bps*	Adjusts the clock rate on the serial interface of the SCI and MCI cards to an acceptable bit rate.

Other Customizing Considerations

In addition to adjusting the SDLLC parameters described in this section, you can improve performance on the connection by adjusting the LLC2 and SDLC parameters described in the chapter "Configuring LLC2 and SDLC Parameters."

For IBM host configuration consider changing the default MAXOUT (window size) value. Widely used installation guides for IBM equipment show a MAXOUT value of 1 in the VTAM-switched major node for the IBM 3174 PU. Changing this value to 7 improves the performance, because VTAM can send seven frames before requiring an acknowledgment.

Monitoring and Maintaining SDLLC Media Translation

To monitor connections using SDLLC media translation, use the following monitoring commands in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# show interfaces`	Displays information about SDLC and LLC2 connections involving interfaces on which SDLLC media translation has been enabled.
Step 2	`Router# show sdllc local-ack`	Displays the current state of any connections using local acknowledgment for LLC2 and SDLLC connections.
Step 3	`Router# show llc2`	Displays information about LLC2 connections involving interfaces on which SDLLC media translation has been enabled.

In **show llc2** command output, look for the LLC2 connections that correspond to the MAC addresses you assigned to the SDLLC interfaces using the **sdllc traddr** command. For information about these commands, see the chapter "LLC2 and SDLC Commands" and "IBM Network Media Translation Commands" in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

QLLC Conversion Configuration Task List

Perform the tasks in the following sections to configure QLLC conversion. The first task is required; all others are optional and depend on your specific needs.

- Enabling QLLC Conversion on a Serial Interface, page 16
- Customizing QLLC Conversion, page 18

See the "QLLC Conversion Configuration Examples" section on page 27 for examples.

Enabling QLLC Conversion on a Serial Interface

The interfaces you configure for QLLC conversion are the serial interfaces that connect to the X.25 network linking the remote devices with which you plan to communicate.

To enable QLLC conversion, you must perform the first of the following tasks. Perform the remaining tasks, as needed.

- Enabling QLLC Conversion on the Appropriate Serial Interfaces, page 16
- Defining the XID Value Associated with an X.25 Device, page 17
- Enabling the Opening of a Connection to the Local Token Ring Device, page 18

Enabling QLLC Conversion on the Appropriate Serial Interfaces

You can enable QLLC conversion on a serial interface to support either a switched virtual circuit (SVC) or a permanent virtual circuit (PVC). The tasks you perform differ somewhat depending on the type of virtual circuit you plan to support on the interface. In either case, first verify that RSRB is enabled by using the following command in privileged EXEC mode:

Command	Purpose
`Router# show configuration`	Ensures that RSRB is enabled on the interfaces.

In the sections for the appropriate serial interfaces of the **show configuration** display, look for one or more **source-bridge remote-peer** entries and a **source-bridge** *rn* entry. For more information about configuring a serial interface for RSRB, see the chapter "Configuring LLC2 and SDLC Parameters."

To enable QLLC conversion to support an SVC, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# x25 map qllc` *virtual-mac-addr x121-addr* [**cud** *cud-value*][*x25-map-options*]	Maps a virtual Token Ring MAC address for the interface to its X.121 address.
Step 2	`Router(config-if)# qllc srb` *virtual-mac-addr srn trn*	Enables the use of QLLC conversion on the interface.

To enable QLLC conversion to support a PVC, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# x25 pvc circuit qllc x121-address [x25-map-options]`	Sets up a PVC for QLLC conversion.
Step 2	`Router(config-if)# qllc srb virtual-mac-addr srn trn`	Enables the use of QLLC conversion on the interface.

To configure QLLC to accept a call from any remote X.25 device, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc accept-all-calls`	Configures QLLC to accept a call from any remote X.25 device.

In a Token Ring or RSRB environment the LAN-attached devices initiate a connection by sending a null XID packet upstream. If the Cisco IOS software forwards this null XID to an X.25-attached FEP, the FEP responds as if it were connecting to an PU2.1 device, and breaks the connection when the PU 2.0 next sends an XID Format 0 Type 2. To resolve this situation and to enable the connection, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc npsi-poll virtual-mac-addr`	Enables connection between a PU 2.0 on the LAN side and a FEP running NPSI on the X.25 side.

The **qllc npsi-poll** command intercepts any null XID packet that the router receives on the LAN interface, and returns a null XID response to the downstream device. It continues to allow XID Format 3 and XID Format 0 packets through the X.25 device.

Defining the XID Value Associated with an X.25 Device

The exchange identification (XID) serves as a password to ensure that only those devices that should communicate with the Token Ring host have that privilege. If the XID is defined in NCP on the host, you must enable the Cisco IOS software to reply (on behalf of the X.25 device) to the Token Ring host's requests for an XID reply. Although the XID value is used to reply to XID requests received on the LLC2 side of the connection, you apply this command on the serial interface defined for X.25. This XID value must match that of IDBLK and IDNUM defined in the NCP.

Note For most QLLC installations, you do not need to define the XID value. You only need to do so if the remote X.25 device is not configured to send its own XID. This is only possible for a device that is attached through a PVC, although most devices that are connected through X.25 send their own XIDs.

To define the XID value associated with an X.25 device, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc xid virtual-mac-addr xid`	Specifies the XID value appropriate for the X.25 device associated with the Token Ring interface.

Enabling the Opening of a Connection to the Local Token Ring Device

If you plan to use SVCs rather than PVCs, you must enable the Cisco IOS software to open a connection to the local Token Ring device on behalf of the remote X.25 device when an incoming call is received. When QLLC conversion is used over an SVC, the remote X.25 device typically initiates the X.25/QLLC session, and the software in turn initiates the LLC2 session.

To enable the software to open a connection to the local Token Ring device, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc partner virtual-mac-addr mac-addr`	Enables the software to open a connection to the local Token Ring device.

Customizing QLLC Conversion

To customize your configuration of QLLC conversion, you can perform one or more of the following tasks in the following sections:

- Enabling QLLC Local Acknowledgment for Remote Source-Route-Bridged Connections, page 18
- Specifying SAP Values Other Than the Default IBM SAP Values, page 19
- Specifying the Largest Packet That Can Be Sent or Received on the X.25 Interface, page 19

Enabling QLLC Local Acknowledgment for Remote Source-Route-Bridged Connections

Enable local acknowledgment when the round-trip time through the TCP/IP network is as large or larger than the LLC2 timeout period.

To enable QLLC local acknowledgment for RSRB connections, use the following global configuration command on the router connected to the X.25 interface and configure the remote peers for local acknowledgment:

Command	Purpose
`Router(config)# source-bridge qllc-local-ack`	Enables QLLC local acknowledgment for remote source-route-bridged connections.

If, for example, Router B with X.25 interface has the IP address *ip1*, and the remote peer (Router A) has the address *ip2*, and they use a virtual ring group *vrg*, then both routers use the following configuration commands:

```
source-bridge ring-group vrg
source-bridge remote-peer vrg tcp ip1 local-ack
```

```
source-bridge remote-peer vrg tcp ip2
```

The configuration for Router B is as follows:

```
source-bridge ring-group vrg
source-bridge remote-peer vrg tcp ip1
source-bridge remote-peer vrg tcp ip2 local-ack
```

This configuration will not affect Router A.

Specifying SAP Values Other Than the Default IBM SAP Values

To use SAP values other than the default IBM SAP values, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc sap virtual-mac-addr ssap dsap`	Specifies a SAP value other than the default IBM SAP value.

Specifying the Largest Packet That Can Be Sent or Received on the X.25 Interface

There are two ways for a packet to become segmented:

- The X.25 software performs the segmentation and the other X.25 station reassembles the packet.
- The QLLC conversion performs SNA header segmentation. In this case, QLLC does not reassemble, but passes smaller SNA segments to the IBM end station.

If the QLLC software does not perform SNA segmentation, then the X.25 software must be capable of performing X.25 segmentation of the largest packet that it can receive from the LLC2 side. This packet can be several thousand bytes long, whereas the typical size for X.25 packets is 1024 bytes or less. (The default is 128 bytes, but that can be overridden with larger values.) The X.25 software, especially in the X.25 attached IBM end station, might not be able to reassemble a very large packet. In this situation, specifying the largest QLLC packet can be useful.

By default, the maximum SNA data unit size established for the virtual circuit is the maximum packet size that can be sent or received on the X.25 interface. If packets received on the LLC2 interface are larger than the largest value allowed on the X.25 connection, they can be segmented by the X.25 software before being sent on the X.25 interface. Moreover, there is no reassembly on receiving packets on the X.25 interface before sending them on the LLC2 interface. Thus, you might need to reconfigure the maximum packet size for the X.25 interface to match that for the LLC2 interface.

When the remote X.25 device has a limit on the maximum total length of recombined X.25 segments it will support, you must ensure the length is not exceeded. For example, a device whose maximum SNA packet size is limited to 265 bytes might not be able to handle a series of X.25 packets that it has to recombine to make a 4-, 8-, or 17-KB SNA packet, such as one often encounters in an LLC2 environment.

You cannot configure the X.25 interface with a larger packet size than the LLC2 interface.

To specify the largest packet that can be sent or received on the X.25 interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# qllc largest-packet virtual-mac-address max-size`	Specifies the largest packet that can be sent or received on the X.25 interface.

Monitoring and Maintaining QLLC Conversion

To monitor connections using QLLC conversion, use the following commands in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# show interfaces serial number`	Displays information about X.25 and LLC2 connections involving interfaces on which QLLC conversion has been enabled.
Step 2	`Router# show qllc`	Displays the current state of any connections using QLLC local acknowledgment.
Step 3	`Router# show llc2`	Displays information about LLC2 connections involving interfaces on which QLLC conversion has been enabled.

SDLLC Configuration Examples

The following sections provide SDLLC configuration examples:

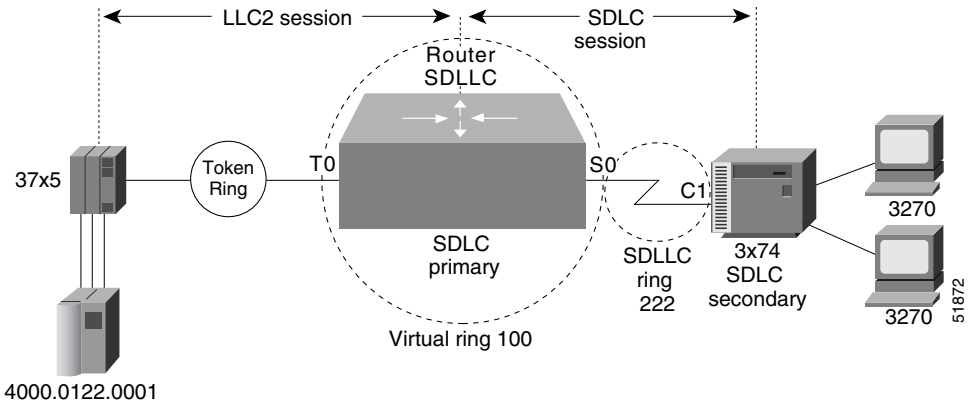
- SDLLC with Direct Connection Example, page 21
- SDLLC with Single Router Using RSRB Example, page 22
- SDLLC with RSRB (Single 3x74) Example, page 23
- SDLLC with RSRB (Multiple 3x74s) Example, page 24
- SDLLC with RSRB and Local Acknowledgment Example, page 25

In the "QLLC Conversion Configuration Examples" section on page 27, refer to the "NCP and VTAM Sysgen Parameters" section on page 31 for sample NCP definitions that the 37x5 FEP in these topologies could use and for sample VTAM definitions that the IBM host in these topologies could use to reflect the routers in the communication path.

SDLLC with Direct Connection Example

Figure 177 shows a router configuration when the router directly connects the Token Ring and the serial line. The Cisco IOS software is configured with SRB.

Figure 177 SDLLC Communication Between a 37x5 and a 3x74 Connected to the Same Router (Direct Connection)



The following configuration enables direct connection:

```
source-bridge ring-group 100
!
interface tokenring 0
 source-bridge 111 1 100
!
interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
 sdllc traddr 0110.2222.3300 222 2 100
 sdllc partner 4000.0122.0001 c1
 sdllc xid c1 1720001
```

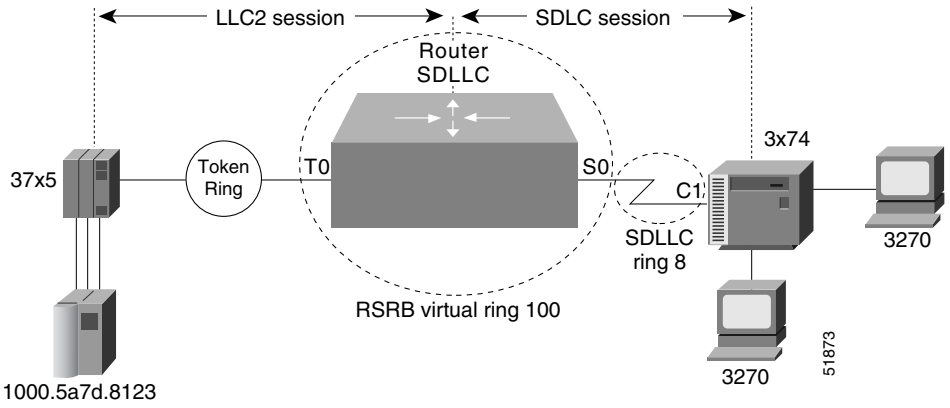
SDLLC with Single Router Using RSRB Example

Figure 178 shows a software configuration in which the router directly connects the Token Ring and the serial line, but uses RSRB to create a virtual ring 100. This configuration has the following characteristics:

- The FEP (37x5) sees C1 3x74 at MAC address 0110.2222.3300
- The RIF from the FEP to the devices would appear as:

  ring 111—bridge 1—ring 100—bridge 1—ring 8

Figure 178 SDLLC with Single Router Using RSRB



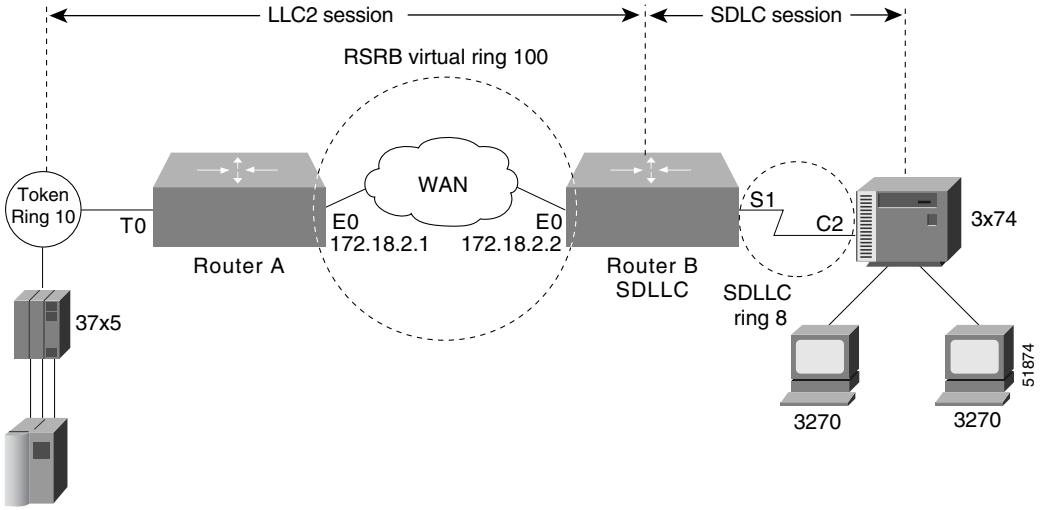
The following sample configuration file is for SDLLC with a single router using RSRB:

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.1.1
source-bridge remote-peer 100 tcp 172.18.2.2

interface tokenring 0
 ip address 172.18.2.2 255.255.255.0
 source-bridge 111 1 100

interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
 sdllc traddr 0110.2222.3300 8 1 100
 sdllc partner 1000.5a7d.8123 c1
 sdllc xid c1 17200c1
```

SDLLC with RSRB (Single 3x74) Example

In Figure 179, SDLLC with RSRB connects a FEP (37x5) and a single 3x74 cluster controller. The host wants to communicate with a single 3174 that its FEP sees on a Token Ring. However, the 3x74 seen by the FEP is in fact SDLC device C1 connected by means of a serial link through a remote router.

Figure 179 SDLLC with RSRB with a Single 3x74



The configuration files for the network shown in Figure 179 follow.

Router A

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.1.1
source-bridge remote-peer 100 tcp 172.18.2.2
!
interface tokenring 0
 ip address 172.18.1.1 255.255.255.0
 source-bridge 10 1 100
!
interface ethernet 0
 ip address 172.18.2.1 255.255.255.0
```

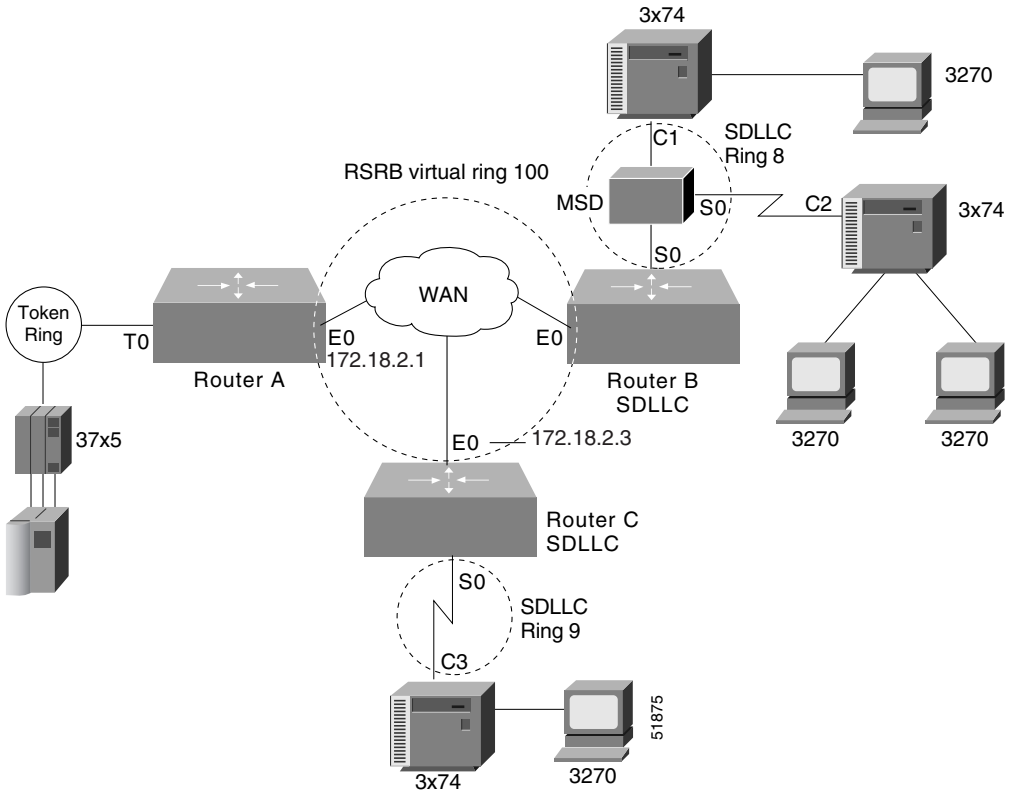
Router B

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.1.1
source-bridge remote-peer 100 tcp 172.18.2.2
!
interface tokenring 0
 ip address 172.18.2.2 255.255.255.0
 source-bridge 1 1 100
!
interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
 sdllc traddr 0110.2222.3300 8 1 100
 sdllc partner 1000.5a7d.8123 c1
 sdllc xid c1 17200c1
```

SDLLC with RSRB (Multiple 3x74s) Example

In the setup shown in Figure 180, Router A needs no SDLLC configuration, Router B has the SDLLC configuration and supports multipoint on the SDLC link with a modem-sharing device, and Router C is also configured with SDLLC. For information about the NCP and VTAM system generation (sysgen) parameters that are used in this configuration, see the "NCP and VTAM Sysgen Parameters" section on page 31. (The notes in the sample configuration files refer to the "Notes" section within the "NCP and VTAM Sysgen Parameters" section on page 31.)

Figure 180 SDLLC with RSRB with Multiple 3x74s



The following configuration files describe the network shown in Figure 180.

Router A

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.2.1
source-bridge remote-peer 100 tcp 172.18.2.2
source-bridge remote-peer 100 tcp 172.18.2.3
!
interface tokenring 0
 ip address 172.18.1.1 255.255.255.0
 source-bridge 10 1 100
!
interface ethernet 0
 ip address 172.18.2.1 255.255.255.0
```

Router B

```
source-bridge ring-group 100
```

```
source-bridge remote-peer 100 tcp 11.108.2.1
source-bridge remote-peer 100 tcp 11.108.2.2
source-bridge remote-peer 100 tcp 11.108.2.3
!
interface ethernet 0
 ip address 172.18.2.2 255.255.255.0
!
interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
 sdlc address c2
 sdllc traddr 0110.2222.3300 7 1 100
 sdllc partner 1000.5a7d.8123 c1
 sdllc partner 1000.5a7d.8123 c2
 sdllc xid c1 17200c1
 sdllc xid c2 17200c2
```

Router C

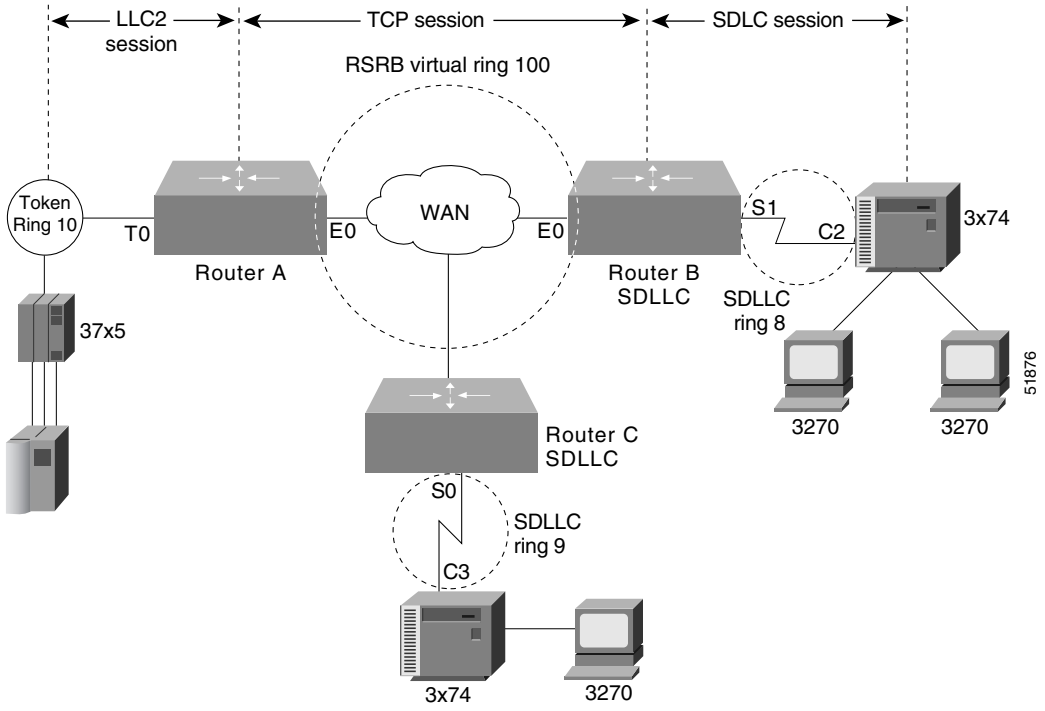
```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.2.1
source-bridge remote-peer 100 tcp 172.18.2.2
source-bridge remote-peer 100 tcp 172.18.2.3
!
interface ethernet 0
 ip address 172.18.2.3 255.255.255.0
!
interface serial 0
 encapsulation sdlc-primary
 sdlc address c3
 sdllc traddr 0110.2222.3300 9 1 100
 sdllc partner 1000.5a7d.8123 c3
 sdllc xid c3 17200c3
```

SDLLC with RSRB and Local Acknowledgment Example

The configuration shown in Figure 181 enables local acknowledgment for Router B, which means that the LLC session terminates at Router A. However, the LLC2 session between Router A and Router C is not locally acknowledged and terminates at Router C.

For information about the NCP and VTAM system generation (sysgen) parameters that are used in this configuration, see the "NCP and VTAM Sysgen Parameters" section on page 31.

Figure 181 SDLLC with RSRB and Local Acknowledgment



The following sample configuration files describe the network shown in Figure 181. (The notes in the sample configuration files refer to the "Notes" section within the "NCP and VTAM Sysgen Parameters" section on page 31.)

Router A

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.2.1
source-bridge remote-peer 100 tcp 172.18.2.2 local-ack
source-bridge remote-peer 100 tcp 172.18.2.3
!
interface tokenring 0
 ip address 172.18.1.1 255.255.255.0
 source-bridge 1 1 100
!
interface ethernet 0
 ip address 172.18.2.1 255.255.255.0
```

Router B

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.2.1 local-ack
source-bridge remote-peer 100 tcp 172.18.2.2
source-bridge remote-peer 100 tcp 172.18.2.3
source-bridge sdllc local-ack
!
interface ethernet 0
 ip address 172.18.2.2 255.255.255.0
!


interface serial 0
 encapsulation sdlc-primary
 sdlc address c1
```

```
 sdllc traddr 4000.3174.0b0d 7 1 100
 sdllc partner 1000.5a7d.8123 c1
 sdllc xid c1 017200c1
!
interface serial 1
 encapsulation sdlc-primary
 sdlc address c2
 sdllc traddr 0110.2222.3200 8 1 100
 sdllc partner 1000.5a7d.8123 c2
 sdllc xid c2 017200c2
```

Router C

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.2.1
source-bridge remote-peer 100 tcp 172.18.2.2
source-bridge remote-peer 100 tcp 172.18.2.3
!
interface ethernet 0
ip address 172.18.2.3 255.255.255.0
!
interface serial 0
 encapsulation sdlc-primary
 sdlc address c3
 sdllc traddr 4000.3174.0c00 9 1 100
 sdllc partner 1000.5a7d.8123 c3
 sdllc xid c3 017200c3
```

QLLC Conversion Configuration Examples

The following sections provide QLLC conversion configuration examples and information:

The examples describe four increasingly complex QLLC conversion topologies and possible software configurations for each. Following the examples are sample NCP definitions that the 37x5 FEP in these topologies could use and VTAM definitions that the IBM host in these topologies could use to reflect the routers in the communication path.

QLLC Conversion Between a Single 37x5 and a Single 3x74 Example

Figure 181, shown previously, illustrates the simplest QLLC conversion topology—a single 37x5 FEP on a Token Ring communicating with a single 3x74 cluster controller across an X.25 network. A router connects the Token Ring to the X.25 network. In Figure 181, notice that the router's X.25 interface is treated as a virtual ring for configuration purposes.

The following configuration file configures the Cisco IOS software to support the network topology shown in Figure 181:

```
source-bridge ring-group 100
!
interface serial 0
 encapsulation x25
 x25 address 31102120100
 x25 map qllc 0100.0000.0001 31104150101
 qllc srb 0100.0000.0001 201 100
!
 ! Allow the 3x74 to initiate the connection.
!
 qllc partner 0100.0000.0001 4000.0101.0132

interface tokenring 0
 source-bridge 1 1 100
```

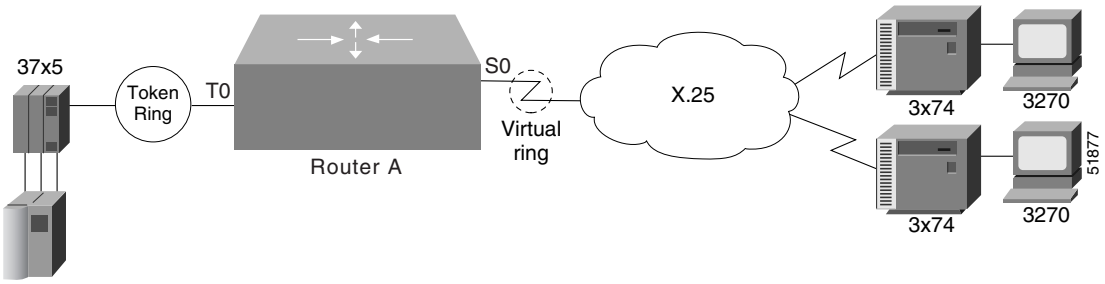
In this configuration file, the **source-bridge ring-group** command defines a virtual ring number 100. The serial 0 interface that connects to the X.25 network is then configured for X.25 DTE operation using the **encapsulation x25** command and assigned the X.121 address of 31102120100 using the **x25 address** command. The **x25 map qllc** command associates the X.121 address of the remote X.25 device (31104150101) with a virtual Token Ring MAC address (0100.0000.0001) the Token Ring device will use to communicate with this remote X.25 device. The **qllc srb** command indicates that the virtual MAC address of the X.25 device will be used to communicate with the real MAC address of the Token Ring device.

The **qllc partner** command enables the software to open a connection to the local Token Ring device at MAC address 4000.0101.0132 on behalf of the remote X.25 device at virtual Token Ring MAC address 0100.0000.0001. The **source-bridge** command configures the router's Token Ring 0 interface for local source-route bridging by associating the router's virtual ring number 100 with the ring number (1) of the local Token Ring and the bridge number (1) that uniquely identifies this bridge interface.

QLLC Conversion Between a Single 37x5 and Multiple 3x74s Example

Figure 182 shows a slightly more complex QLLC conversion topology. The same 37x5 FEP on a Token Ring connects through a router to an X.25 network, but communicates with multiple 3x74 cluster controllers through X.25.

Figure 182 QLLC Conversion Between a Single 37x5 and Multiple 3x74s



The following configuration file configures the Cisco IOS software to support the network topology shown in Figure 182:

```
source-bridge ring-group 100
!
interface serial 0
 encapsulation x25
 x25 address 3137005469
!
 ! configure the first 3174
!
 x25 map qllc 0000.0cff.0001 31370054111
!
 ! 1001 - virtual ring used by all QLLC devices
 ! 100 - the virtual ring group
!
 qllc srb 0000.0cff.0001 1001 100
 qllc partner 0000.0cff.0001 4000.1160.0000
 qllc xid 0000.0cff.0001 01710017
!
 ! configure the second 3174
!
 x25 map qllc 0000.0cff.0002 313700543247
!
 ! 1001 - virtual ring used by all QLLC devices
 ! 100 - the virtual ring group
!
 qllc srb 0000.0cff.0002 1001 100
 qllc partner 0000.0cff.0002 4000.1160.0000
 qllc xid 0000.0cff.0002 01710017
!
interface tokenring 0
!
 ! Because this is a real bridge, we have to define the way it
 ! bridges to the QLLC virtual ring.
!
 source-bridge 1 1 100
 source-bridge spanning
```

QLLC Conversion Between Multiple 37x5s and Multiple 3x74s Example

In the following example, two 3x74s on a Token Ring each attach to a different 37x5 on the other side of an X.25 network. Only one Token Ring interface is used. Do not create a bridge from the QLLC virtual ring (1001) to the physical Token Ring (1). Instead, define a virtual ring group (for example, 100).

```
interface serial 0
 encapsulation x25
 x25 address 3137005469
!
 ! configure the router for the first 3x74
!
 x25 map qllc 0000.0cff.0001 31370054111
!
 ! 1001 - virtual ring used by all QLLC devices
 ! 1 - the local Token Ring number
!
 qllc srb 0000.0cff.0001 1001 1
 qllc partner 0000.0cff.0001 4000.1160.0000
!
 ! configure the router for the second 3x74
!
 x25 map qllc 0000.0cff.0002 31370053247
!
 ! 1001 - virtual ring used by all qllc devices
 ! 1 - the local Token Ring number
!
 ! Note that the partner's MAC address and XID are different from
 ! those in the first 3x74.
!
 qllc srb 0000.0cff.0001 1001 1
 qllc partner 0000.0cff.0002 4000.1161.1234
!
interface tokenring 0
!
 ! Because this is a real bridge, we have to define the way it bridges
 ! to the QLLC virtual ring.
!
 source-bridge 1 1 1001
 source-bridge spanning
```

QLLC Conversion Between a Single 37x5 and Multiple 3x74s Across an Arbitrary WAN Example

Figure 182 includes an added arbitrary WAN in the communication path between the 37x5 FEP and the multiple 3x74 cluster controllers. The arbitrary WAN can be a multihop network, whereas QLLC conversion treats the X.25 network as a single-hop network.

In Figure 182, notice that the arbitrary WAN and the routers on either side of it form a single virtual ring, as configured using the **source-bridge ring-group** global configuration command.

In this configuration file, Router A uses an IP address of 172.18.2.2 and its Token Ring interface is attached to Token Ring 1. Because Router A connects to the Token Ring, it does not need to be configured for QLLC conversion. Router B, configured for QLLC conversion because it connects directly to the X.25 network through its serial interface, uses an X.121 address of 31102120100 and an IP address of 172.18.1.1. The 37x5 device uses a MAC address of 4000.0101.0132. The virtual MAC address of 0100.0000.0001 has been assigned to the 3x74 device.

Router A

The following configuration file configures the Router A in Figure 182:

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.1.1 local-ack
source-bridge remote-peer 100 tcp 172.18.2.2 local-ack
!
interface ethernet 0
 ip address 131.108.3.3 255.255.255.0
!
interface tokenring 0
 ip address 172.18.2.2 255.255.255.0
 source-bridge 1 1 100
 source-bridge spanning
```

Router B

The following configuration file configures the Router B in Figure 182:

```
source-bridge ring-group 100
source-bridge remote-peer 100 tcp 172.18.1.1 local-ack
source-bridge remote-peer 100 tcp 172.18.2.2 local-ack
source-bridge qllc-local-ack
!
interface serial 0
 encapsulation x25
 x25 address 31102120100
 x25 map qllc 0100.0000.0001 31104150101
 x25 map qllc 0100.0000.0002 31104150102
 qllc srb 0100.0000.0001 201 100
 qllc srb 0100.0000.0002 201 100
!
 ! Allow the 3174 to initiate the connection.
!
 qllc partner 0100.0000.0001 4000.0101.0132
 qllc partner 0100.0000.0002 4000.0101.0132
!
interface ethernet 0
 ip address 172.18.1.1 255.255.255.0
```

NCP and VTAM Sysgen Parameters

The sample system generation (sysgen) parameters in this section show typical NCP and VTAM values that correspond with configurations for Router A, Router B, and Router C in Figure 180, Figure 181, and Figure 182. Figure 180 and Figure 181 show SDLLC media translation and Figure 182 shows QLLC conversion.

IBM's ACF/NCP uses a function called NTRI (NCP/Token Ring Interconnection) to support Token Ring-attached SNA devices. NTRI also provides translation from Token Ring-attached SNA devices (Physical Units) to switched (dial-up) devices. VTAM provides the resolution for these devices in a Switched Major Node. VTAM treats these devices on NTRI logical lines as switched devices. (For more information consult IBM documentation *NCP/SSP/EP Resource Definition Reference*, SC30-3448-04.)

Using SDLLC, the Cisco IOS software translates SDLC leased line protocol into Token Ring LLC2 protocol, then the NTRI function in ACF/NCP translates Token Ring LLC2 protocol into an SNA switched protocol.

NCP Generation Definitions

```
************************************************************************
```

```
***               SAMPLES BASED ON ACF/NCP V5 R4.
***             NOT ALL NCP PARAMETERS ARE SHOWN
*****************************************************************
*
*****************************************************************
*   OPTIONS DEFINITION STATEMENT
*****************************************************************
NCPOPTOPTIONSNEWDEFN=YESNTRI GENERATION, MUST BE FIRST STMT
*
*****************************************************************
*   BUILD MACRO
*****************************************************************
NCPBUBUILDLOCALTO=1.5,NTRI ACK TIMER FOR LOCAL TOKEN RINGS
        REMOTTO=2.5,NTRI ACK TIMER FOR REMOTE TOKEN RINGS
            USED IN SDLLC CONFIGURATIONS, NOTE 1.
*
*****************************************************************
*   DYNAMIC RECONFIGURATION POOL SPACE
*****************************************************************
DRPOOLLUDRPOOLNUMTYP2=50RESERVE 50 LUS ON PU. T2 PUS
*
*****************************************************************
*   PHYSICAL GROUP FOR NTRI TIC #1, DEFINITIONS FOR THE TOKEN RING
*   ADAPTER TO ESTABLISH PHYSICAL CONNECTIVITY
*****************************************************************
EPHYGGROUPECLTYPE=PHYSICAL
*
EPHYLLINEADAPTER=TIC2,TYPE OF ADAPTER
        ADDRESS=(16,FULL),INTERNAL FEP TIC ADDRESS
        PORTADD=0,
        LOCADD=10005a7d8123,TIC ADDRESS,
        RCVBUFC=1440,
        MAXTSL=2012,
        TRSPEED=16TOKEN RING SPEED
*
EPHYPUPU
*
EPHYLULUISTATUS=INACTIVE
*
*****************************************************************
*   NTRI PERIPHERAL LOGICAL LINE GROUP, LINE AND PU PAIRS ARE
*   GENERATED BY THE AUTOGEN PARAMETER.
*****************************************************************
ELOGGGROUPECLTYPE=LOGICAL,
        PHYPORT=0,
        CALL=INOUT,
        AUTOGEN=3ONE PER SDLLC CONTROLLER,
*****************************************************************
```

VTAM Definitions

```
*****************************************************************
*   VTAM SWITCHED MAJOR NODE, BASED ON ACF/VTAM V3 R4.
*   THE CODING BELOW SUPPORTS DIAL IN OPERATION ONLY. TYPICALLY,
*   NTRI IMPLEMENTATIONS USE ONLY DIAL IN. IF DIAL OUT FROM AN
*   APPLICATION IS REQUIRED, PATH MACROS MUST BE USED. CONSULT
*   THE APPROPRIATE VTAM INSTALLATION REFERENCE MANUAL.
*****************************************************************
VSWITCHVBUILDTYPE=SWNET
*
VPU1PU  ADDR=13,COULD BE ANYTHING (NOT USED)
        IDBLK=017,XID PARM,
        IDNUM=200c1,XID PARM,
        MAXOUT=7,
```

```
        MAXDATA=265,
        MODETAB=AMODETAB,
        DLOGMOD=US327X,
        PUTYPE=2,
        USSTAB=USS327X
*
VLU1ALU LOCADDR=2,
VLU1BLU LOCADDR=3
*
VPU2PU  ADDR=13,COULD BE ANYTHING (NOT USED)
        IDBLK=017,XID PARM,
        IDNUM=200c2,XID PARM,
        MAXOUT=7,
        MAXDATA=265,
        MODETAB=AMODETAB,
        DLOGMOD=US327X,
        PUTYPE=2,
        USSTAB=USS327X
*
VLU2ALU LOCADDR=2,
VLU2BLU LOCADDR=3
*
VPU3PU  ADDR=13,COULD BE ANYTHING (NOT USED)
        IDBLK=017,XID PARM,
        IDNUM=200c3,XID PARM,
        MAXOUT=7,
        MAXDATA=265,
        MODETAB=AMODETAB,
        DLOGMOD=US327X,
        PUTYPE=2,
        USSTAB=USS327X
*
VLU3ALU LOCADDR=2,
VLU3BLU LOCADDR=3
*
```

Notes

In these sample definitions:

1. REMOTTO is the NCP's T1 timer for remote Token Rings. All connections use RIF information and therefore look like remote Token Ring devices. The default is 2.5 seconds, which is adequate for most situations; however, when slow-speed links are used, this parameter should be reviewed to ensure enough time for link-level acknowledgments.
2. The LOCADD parameter defines the locally administered address of the TIC in the NCP. The Cisco IOS software, configured for SDLLC, will insert this address as the 802.5 destination address field in TEST and XID frames to establish connectivity and then in data frames during the session. The **sdllc partner** and **qllc partner** commands define this connection in the Cisco IOS software. Each SDLC control unit is defined with an **sdllc partner** or **qllc partner** command.
3. The AUTOGEN parameter specifies the number of LINE and PU pairs that are automatically generated by Network Definition Facility (NDF). Each controller requires a LINE and PU definition in the ELCTYPE LOGICAL group. These represent control block space in the NCP simulating switched line as described earlier.
4. The IDBLK and IDNUM parameters in VTAM are used to identify incoming connection requests. IDBLK is typically unique for each type of IBM device. IDNUM is any five hexadecimal digit combination. The Cisco routers configured for SDLLC or QLLC conversion must associate an IDBLK/IDNUM combination with a controller by using the **sdllc xid** or **qllc xid** command. If not using the **qllc xid** command, then IDBLK/IDNUM must agree with the values of the X.25 attached devices. During activation, an XID will be sent to the NCP containing the specific IDBLK/IDNUM. NCP will send these values to VTAM in an SNA command called REQCONT. VTAM will search its switched major nodes to find a match. If found, VTAM will establish sessions with the device by sending activation commands (ACTPU, ACTLUs).

Configuring SNA Frame Relay Access Support

This chapter describes Frame Relay Access Support (FRAS) for Systems Network Architecture (SNA) devices. It also explains how to configure FRAS and how to use a FRAS host to connect Cisco Frame Relay Access Devices (FRADs) to channel-attached mainframes, LAN-attached front-end processors (FEPs), and LAN-attached AS/400s through a Cisco router.

For a complete description of the FRAS commands in this chapter, refer to the "SNA Frame Relay Access Support Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of specific commands, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

FRAS, the Cisco IOS software allows branch SNA devices to connect directly to a central site FEP over a Frame Relay network. FRAS converts LAN or Synchronous Data-Link Control (SDLC) protocols to a Frame Relay format understood by the Network Control Program (NCP) that runs in an FEP. The Cisco IOS software and the NCP support two frame formats:

- RFC 1490 routed format for LLC2, specified in the FRF.3 Agreement from the Frame Relay Forum and known in NCP literature as Frame Relay Boundary Network Node (BNN) support. Support for this feature requires NCP 7.1 or higher.

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

- RFC 1490 802.5 source-route bridged format, known in NCP literature as Frame Relay Boundary Access Node (BAN) support. Support for this feature requires NCP 7.3 or higher.

Management service point support in FRAS allows the SNA network management application, NetView, to manage Cisco routers over the Frame Relay network as if it were an SNA downstream PU.

FRAS provides dial backup over RSRB in case the Frame Relay network is down. While the backup Public Switched Telephone Network (PSTN) is being used, the Frame Relay connection is tried periodically. As soon as the Frame Relay network is up, it will be used.

RFC 1490 Routed Format for LLC2 (BNN)

RFC 1490 specifies a standard method of encapsulating multiprotocol traffic with data link (Level 2 of the OSI model) framing. The encapsulation for SNA data is specified in the FRF.3 Agreement.

The Frame Relay encapsulation method is based on the RFC 1490 frame format for "user-defined" protocols using Q.933 NLPID, as illustrated in Figure 183.

Figure 183 Frame Relay Encapsulation Based on RFC 1490

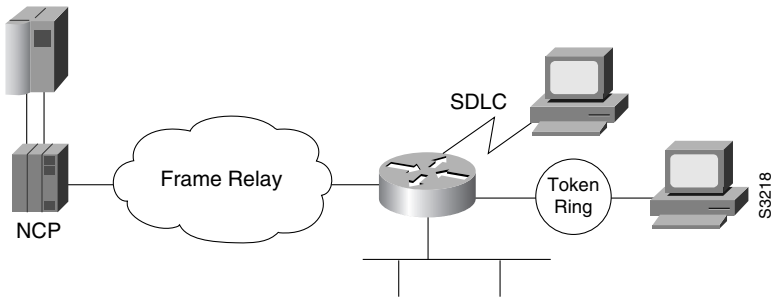
DLCI	Control	NLPID	L2	L3	DSAP	Control	F
Q.922 address	0x30	Q.933 0x08	Protocol ID 0x4c (802.2) 0x08	Protocol ID	SSAP		C S

51911

Note The protocol ID for SNA subarea FID4 is 0x81. The protocol ID for SNA subarea FID2 is 0x82. The protocol ID for APPN FID2 is 0x83.

FRAS allows the router acting as a FRAD to take advantage of the SNA BNN support for Frame Relay provided by ACF/NCP 7.1 and OS/400 V2R3. Downstream PU 2.0 and PU 2.1 devices can be attached to the router through SDLC, Token Ring, or Ethernet links. The router acting as a FRAD is connected to the Network Control Program (NCP) or AS/400 through a public or private Frame Relay network, as illustrated in Figure 184.

Figure 184 SNA BNN Support for Frame Relay



The frame format that communicates across the Frame Relay BNN link is defined in RFC 1490 for routed SNA traffic. From the perspective of the SNA host (for example an NCP or AS/400), the Frame Relay connection is defined as a switched resource similar to a Token Ring BNN link. Because the frame format does not include link addresses to allow the NCP to distinguish among SNA devices on the same permanent virtual circuit, Cisco supports SAP multiplexing, which allows you to configure unique LLC2 SAPs for each downstream SNA device so that they can share a single permanent virtual circuit to an FEP.

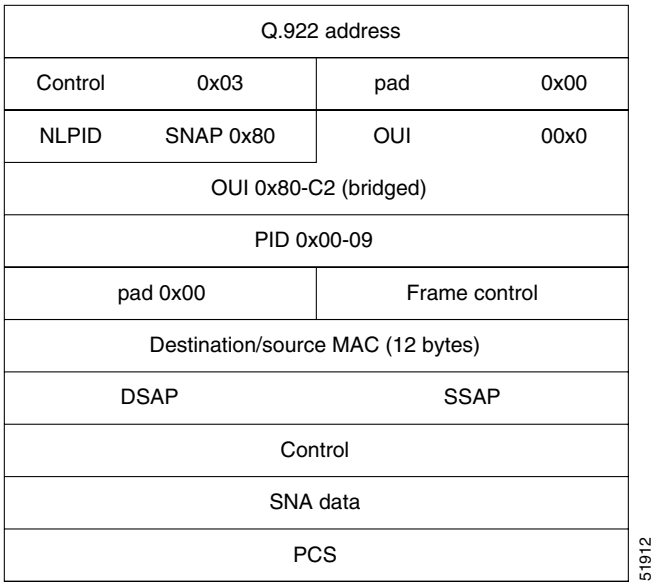
The Cisco IOS software is responsible for terminating the local data-link control frames (such as SDLC and Token Ring frames) and for modifying the data-link control frames to 802.2 compliant LLC frames. The LLC provides a reliable connection-oriented link layer transport required by SNA. (For example, 802.2 LLC is used to provide link-layer acknowledgment, sequencing, and flow control.)

The Cisco IOS software encapsulates these 802.2 LLC frames according to the RFC 1490 format for SNA traffic. The frames are then forwarded to the SNA host on a Frame Relay permanent virtual circuit (PVC). In the reverse direction, the software is responsible for de-encapsulating the data from the Frame Relay PVC, and for generating and sending the appropriate local data-link control frames to the downstream devices.

RFC 1490 Bridged Format for LLC2 (BAN)

BAN provides functionality similar to BNN except that it uses a bridged frame format, as illustrated in Figure 185.

Figure 185 RFC 1490 Bridged Frame Format



Because it includes the MAC header information in every frame, BAN supports multiple SNA devices sharing a single permanent virtual circuit without requiring SAP multiplexing. BAN also supports load balancing across duplicate data-link connection identifiers to the same or different FEPs at the data center to enhance overall availability. BAN works for devices attached by either Token Ring or Ethernet.

SNA FRAS Configuration Task List

To configure FRAS, perform the tasks described in the following sections:

- Configuring FRAS BNN Statically, page 5
- Configuring FRAS BNN Dynamically, page 5
- Configuring FRAS BAN Support, page 6
- Configuring SRB over Frame Relay, page 6

- Configuring FRAS Congestion Management, page 7

- Configuring FRAS DLCI Backup, page 7
- Configuring Frame Relay RSRB Dial Backup, page 8
- Configuring Frame Relay DLSw+ Dial Backup, page 8

To configure the FRAS host, see the "Configuring FRAS Host" section on page 11. For configuration examples, see the "FRAS and FRAS Host Configuration Examples" section on page 15.

Configuring FRAS BNN Statically

To configure FRAS BNN statically, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# fras map llc mac-address lan-lsap lan-rsap serial port frame-relay dlci fr-lsap fr-rsap [pfid2 | afid2 | fid4]`	Associates an LLC connection with a Frame Relay DLCI.
`Router(config-if)# fras map sdlc sdlc-address serial port frame-relay dlci fr-lsap fr-rsap [pfid2 | afid2 | fid4]`	Associates an SDLC link with a Frame Relay DLCI.

In this implementation, you configure and define each end station MAC and SAP address pair statically.

Because Frame Relay itself does not provide a reliable transport as required by SNA, the RFC 1490 support of SNA uses LLC2 as part of the encapsulation to provide link-level sequencing, acknowledgment, and flow control. The serial interface configured for Internet Engineering Task Force (IETF) encapsulation (RFC 1490) accepts all LLC2 interface configuration commands.

Configuring FRAS BNN Dynamically

To configure FRAS BNN dynamically, use one of the following commands in interface configuration mode, as needed:

Command	Purpose
`Router(config-if)# fras map llc lan-lsap serial interface frame-relay dlci dlci fr-rsap`	Associates an LLC connection with a Frame Relay DLCI.
`Router(config-if)# fras map sdlc sdlc-address serial port frame-relay dlci fr-lsap fr-rsap [pfid2 | afid2 | fid4]`	Associates an SDLC link with a Frame Relay DLCI.

When you associate an LLC connection with a Frame Relay DLCI, the router "learns" the MAC/SAP information as it forwards packets to the host. The FRAS BNN feature provides seamless processing at the router regardless of end station changes. End stations can be added or deleted without reconfiguring the router.

When you associate an SDLC link with a Frame Relay DLCI, you configure and define each end station MAC and SAP address pair statically.

Because Frame Relay itself does not provide a reliable transport as required by SNA, the RFC 1490 support of SNA uses LLC2 as part of the encapsulation to provide link-level sequencing, acknowledgment, and flow control. The serial interface configured for IETF encapsulation (RFC 1490) can take all LLC2 interface configuration commands.

Configuring FRAS BAN Support

To configure Frame Relay BAN, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# fras ban local-ring bridge-number ring-group ban-dlci-mac dlci dlci#1 [dlci#2 ... dlci#5] [bni mac-addr]`	Associates a bridge to the Frame Relay BAN.

BAN simplifies router configuration when multiple LLC sessions are multiplexed over the same DLCI. By comparison, SAP multiplexing requires static definitions and maintenance overhead. By using BAN, the Token Ring MAC address is included in every frame to uniquely identify the LLC session. Downstream devices can be dynamically added and deleted with no configuration changes required on the router.

Configuring SRB over Frame Relay

To configure SRB over Frame Relay, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# interface serial number`	Specifies the serial port.
Step 2	`Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 3	`Router(config-if)# interface serial slot/port.subinterface-number point-to-point`	Configures a Frame Relay point-to-point subinterface.
Step 4	`Router(config-if)# frame-relay interface-dlci dlci ietf`	Configures a DLCI number for the point-to-point subinterface.
Step 5	`Router(config-if)# source-bridge source-ring-number bridge-number target-ring-number conserve-ring`	Assigns a ring number to the Frame Relay permanent virtual circuit.

Cisco IOS software offers the ability to encapsulate source-route bridging traffic using RFC 1490 Bridged 802.5 encapsulation. This provides SRB over Frame Relay functionality. This SRB over Frame Relay feature is interoperable with other vendors' implementations of SRB over Frame Relay and with some vendors' implementations of FRAS BAN.

SRB over Frame Relay does not support the following Cisco IOS software functions:

- Proxy explorer
- Automatic spanning tree
- LAN Network Manager

Configuring FRAS Congestion Management

FRAS provides a congestion control mechanism based on the interaction between congestion notification bits in the Frame Relay packet and the dynamic adjustment of the LLC2 send window. This window shows the number of frames the Cisco IOS software can send before waiting for an acknowledgment. The window size decreases with the occurrence of backward explicit congestion notification (BECN) and increases when no BECN frames are received.

To configure congestion management, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# llc2 local-window` *packet-count*	Specifies the maximum window size for each logical connection.
Step 2	`Router(config-if)# llc2 dynwind` [**nw** *nw-number*] [**dwc** *dwc-number*]	Enables the dynamic window flow-control mechanism.

You can enable the dynamic window mechanism only if you are using Frame Relay IETF encapsulation.

Configuring FRAS DLCI Backup

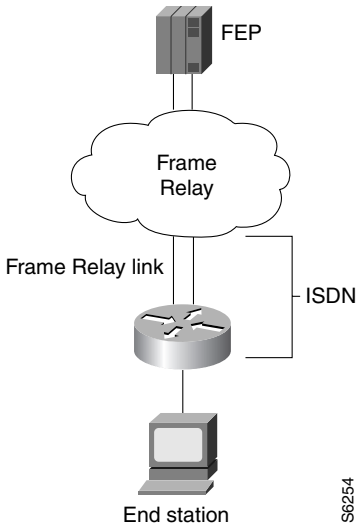
To configure FRAS DLCI backup, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# fras ddr-backup interface` *interface dlci-number*	Specifies an interface to be used for the backup connection and indicate the DLCI number of the session.

FRAS DLCI backup is an enhancement to Cisco's FRAS implementation that lets you configure a secondary path to the host to be used when the Frame Relay network becomes unavailable. When the primary Frame Relay link to the Frame Relay WAN fails, the FRAS DLCI backup feature causes the router to reroute all sessions from the main Frame Relay interface to the secondary interface. The secondary interface can be either serial or ISDN and must have a data-link connection identifier (DLCI) configured.

Figure 186 illustrates Frame Relay backup over an ISDN connection.

Figure 186 FRAS DLCI Backup over ISDN



Note This feature provides backup for the local end of the Frame Relay connection, not the complete end-to-end connection.

Configuring Frame Relay RSRB Dial Backup

When the Frame Relay network is down, the Cisco IOS software checks whether the dial backup feature is configured for the particular DLCI number. If it is configured, the software removes the FRAS to the downstream device connection and establishes the RSRB to this downstream device connection.

To configure RSRB dial backup, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# fras backup rsrb` *vmacaddr local-ring-number target-ring-number host-mac-address*	Activates Frame Relay RSRB dial backup.

Configuring Frame Relay DLSw+ Dial Backup

The FRAS dial backup over DLSw+ feature provides a secondary path that is used when the Frame Relay network becomes unavailable. If preconfigured properly, when the primary link to the Frame Relay WAN fails, FRAS dial backup over DLSw+ feature moves existing sessions to the alternate link automatically. When the primary link is restored, existing sessions are kept on the backup connection so they can be moved non-disruptively to the primary link at the user's discretion.

To enable FRAS dial backup over DLSw+, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# fras backup dlsw` *virtual-mac-address target-ring-number host-mac-address* [**retry** *number*]	Configures an auxiliary (backup) route between the end stations and the host for use when the DLCI connection to the Frame Relay network is lost.

Figure 187 shows a Frame Relay network with FRAS dial backup over DLSw+.

Figure 187 FRAS Dial Backup over DLSw+

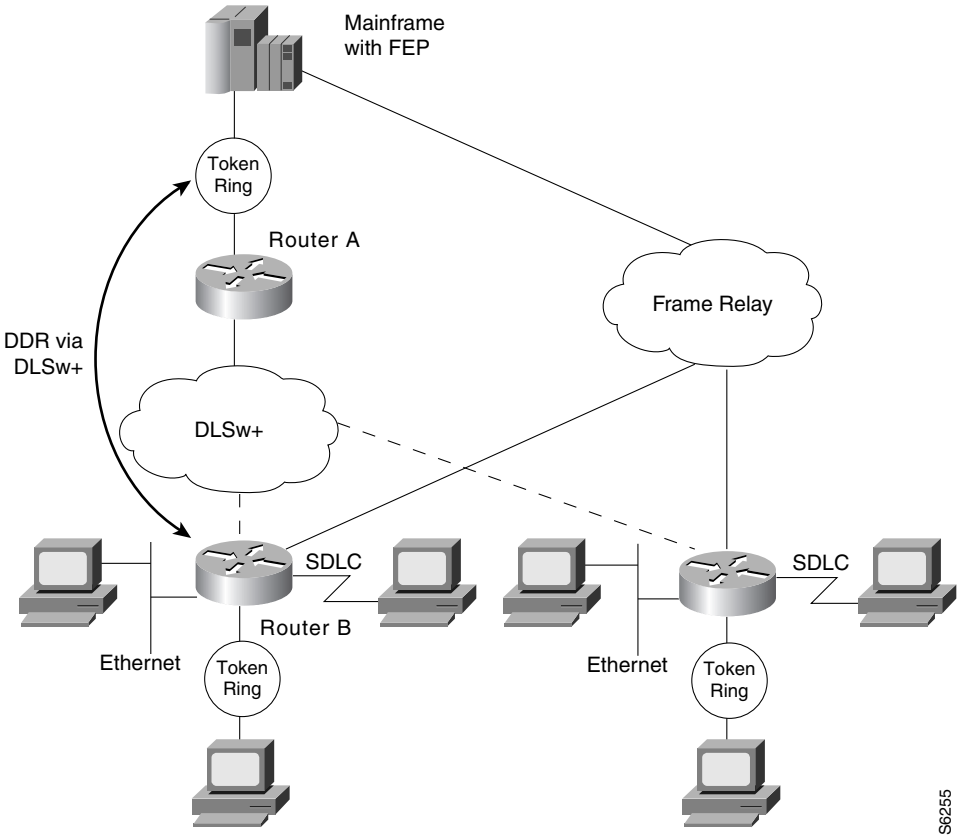
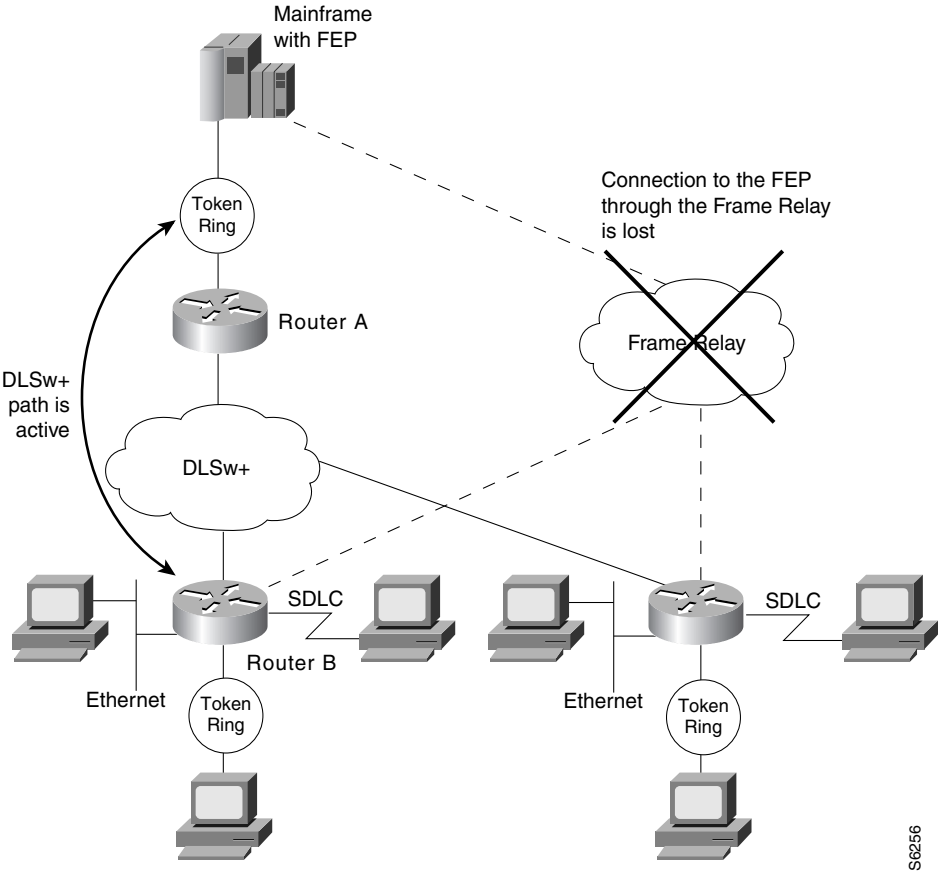


Figure 188 shows the active FRAS dial backup over DLSw+ when the Frame Relay connection to the NCP is lost.

Figure 188 FRAS Dial Backup over DLSw+ when Frame Relay is Unavailable



Monitoring and Maintaining FRAS

To display information about the state of FRAS, use the following command in privileged EXEC mode:

Command	Purpose
`Router# show fras`	Displays the mapping and connection state of the FRAS.

Configuring FRAS Host

The FRAS host provides a scalable and efficient solution for SNA FRAD access to channel-attached hosts and to LAN-attached hosts. The FRAS host function operates in two modes, which are documented in the following sections:

- FRAS Host LLC2 Passthrough, page 11—In this mode, the LLC2 sessions are not locally terminated in the router's LLC2 stack. This is the recommended solution if your scenario includes a Channel Interface Processor (CIP) interface to the mainframe.
- FRAS Host LLC2 Local Termination, page 12—In this mode, the LLC2 sessions are locally terminated in the router's LLC2 stack. This is the recommended solution if either of the following is true:
  - Your scenario includes a LAN-attached AS/400 or mainframe.
  - Your scenario includes conversion from RFC1490 encapsulation to DLSw+ encapsulation.

FRAS Host LLC2 Passthrough

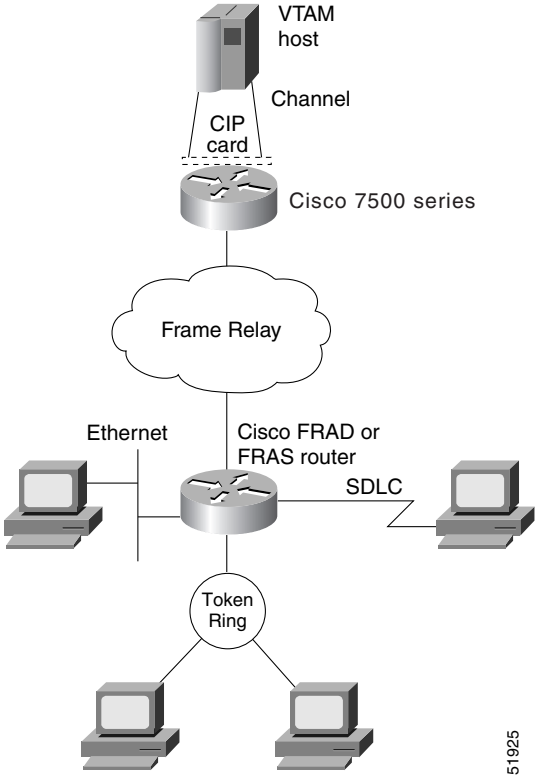
The FRAS host LLC passthrough feature combines with a CIP-attached Cisco router's high-speed channel access to provide FEP-class performance at a fraction of what it would cost to achieve similar functionality using a FEP. If the CIP SNA feature is used to interface with the mainframe, then FRAS host LLC2 passthrough mode is the recommended solution. In this topology the LLC2 passthrough solution to the CIP-SNA LLC2 stack provides better performance, is more robust, and responds well to different types of congestion.

To prevent LLC2 session timeout, LLC2 characteristics (windows and timers) may be tuned on the CIP internal LAN adapter. The CIP/SNA LLC2 stack reacts to congestion by dynamically adjusting its LLC2 send window for that LLC2 session in response to dropped frames.

With the FRAS host LLC passthrough feature, you gain performance benefits of a channel attachment without FEP upgrades such as the addition of a Frame Relay interface, an upgrade to NCP (with its associated increase in monthly charges), and a possible increase in system memory.

Figure 189 illustrates Cisco FRAD access to a mainframe through a channel-attached Cisco router.

Figure 189 Cisco FRAD Access to a Mainframe through a Cisco 7500



FRAS Host LLC2 Local Termination

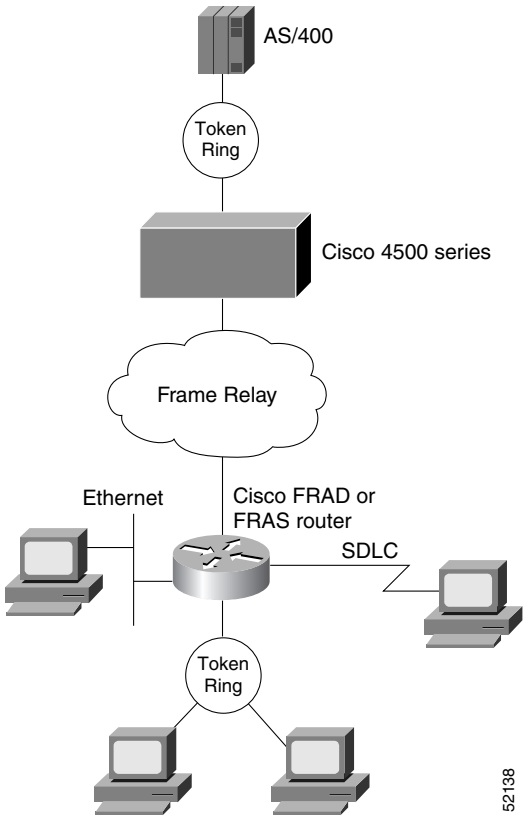
If the FRAS host feature is used to allow remote FRADs to communicate with a LAN-attached IBM 3745 or AS/400, then LLC2 termination via DLSw+ local switching is the recommended solution. With this approach, the LLC2 sessions are terminated at the Route Processor. To prevent LLC2 session timeout, LLC2 characteristics (windows and timers) may be tuned on the virtual Token Ring interface. If the dynamic window algorithm is enabled on the virtual Token Ring interface, LLC2 local termination will react to congestion by dynamically adjusting its LLC2 send window in response to occurrence of Frame Relay BECN.

When you use the FRAS host LLC2 local termination feature on a Token Ring-attached FEP, the FRAS host Cisco router shields the FEP from having to manage the interface to the Frame Relay network. This avoids interface, memory, and NCP upgrades. The FRAS host Cisco router simply provides LLC2 sessions to the FEP over the LAN.

If used in an environment with AS/400s, FRAS host LLC2 local termination provides an even more valuable function. The Cisco FRAS host router offloads the management of the Frame Relay connections from the AS/400. This reduces AS/400 system hardware requirements and frees AS/400 CPU cycles for user applications.

Figure 190 illustrates Cisco FRAD access to a LAN-attached SNA host through a Cisco router.

Figure 190 Cisco FRAD Access to a LAN-Attached AS/400 through a Cisco 4500



Congestion Management

Both passthrough and local acknowledgment environments support frame discard eligibility (DE) for additional congestion management. In both environments, you can further tune the interface to the Frame Relay network by taking advantage of the Cisco IOS Frame Relay features. Taken together, these features increase overall throughput dramatically by comparison to generic FRADs, which typically cannot use the network with the same degree of efficiency.

FRAS Host Configuration Task List

To configure the FRAS host migration feature, perform the tasks in the following sections:

- Creating a Virtual Token Ring Interface, page 14
- Configuring Source-Route Bridging on the Virtual Token Ring Interface, page 14
- Accepting Default LLC2 Passthrough or Enabling LLC2 Local Termination, page 14
- Enabling the FRAS Host Feature for BAN or BNN, page 15
- Monitoring LLC2 Sessions Using FRAS Host, page 15

See the "FRAS and FRAS Host Configuration Examples" section on page 15 for examples.

Creating a Virtual Token Ring Interface

To configure a virtual Token Ring interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# interface virtual-tokenring` *number*	Configures a virtual Token Ring interface.

Configuring Source-Route Bridging on the Virtual Token Ring Interface

To configure SRB on the Token Ring interface, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group` *ring-group virtual-mac-address*	Enables local SRB.
Step 2	`Router(config)# source-bridge` *local-ring bridge-number target-ring*	Enables FRAS host traffic to access the SRB domain.

Note If you are using LLC2 passthrough with an Ethernet-attached host, you must configure the Cisco source-route translational bridging (SR/TLB) feature.

Accepting Default LLC2 Passthrough or Enabling LLC2 Local Termination

LLC2 passthrough is the default operational mode for all FRAS host connections that use a virtual Token Ring interface. You do not need to perform any configuration to accept the default LLC2 passthrough mode.

To enable LLC2 local termination for FRAS host connections using the virtual Token Ring, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# dlsw local-peer`	Defines the parameters of the DLSw+ local peer.
Step 2	`Router(config)# fras-host dlsw-local-ack`	Enables LLC2 local termination for FRAS host connections.

Enabling the FRAS Host Feature for BAN or BNN

To enable the FRAS host for BAN or BNN, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# fras-host bnn` *interface* **fr-lsap** *sap* **vmac** *virt-mac* **hmac** *hmac* [**hsap** *hsap*]	Configures the FRAS host for BNN.
Step 2	`Router(config-if)# fras-host ban` *interface* **hmac** *hmac* [**bni** *bni-mac*]	Configures the FRAS host for BAN.

Monitoring LLC2 Sessions Using FRAS Host

To display the status of LLC2 sessions using FRAS host, use the following command in privileged EXEC mode:

Command	Purpose
`Router#` **show fras-host** [*interface*] [**dlci** *dlci-num*] [**detail**]	Displays the status of LLC2 sessions using FRAS host.

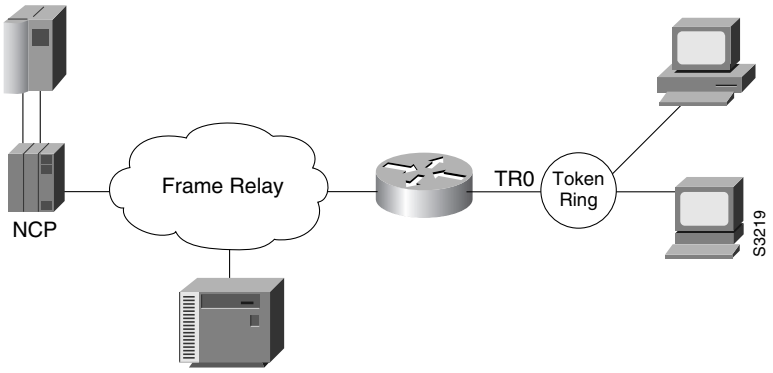
FRAS and FRAS Host Configuration Examples

The following sections provide both FRAS and FRAS host configuration examples:

LAN-Attached SNA Devices Example

Figure 191 illustrates the configuration of SNA devices attached to a LAN.

Figure 191 LAN-Attached SNA Devices



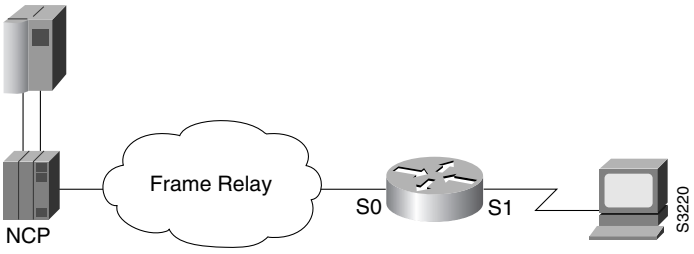
The configuration for the network shown in Figure 191 is as follows:

```
interface tokenring 0
 no ip address
 no keepalive
 ring-speed 16
 fras map llc 0800.5a8f.8802 4 4 serial 0 frame-relay 200 4 4
!
interface serial 0
 mtu 2500
 no ip address
 encapsulation frame-relay IETF
 keepalive 12
 frame-relay lmi-type ansi
 frame-relay map llc2 200
```

SDLC-Attached SNA Devices Example

Figure 192 illustrates the configuration of SDLC-attached SNA devices.

Figure 192 SDLC-Attached SNA Devices



The configuration file for the network shown in Figure 192 is as follows:

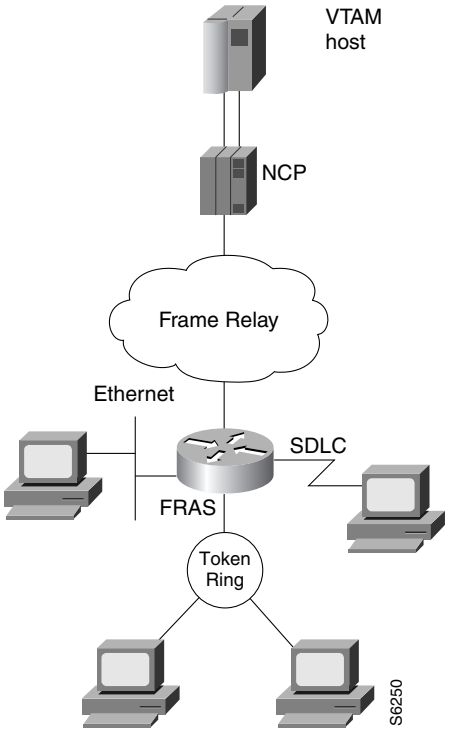
```
interface serial 1
 no ip address
 encapsulation sdlc
 no keepalive
```

```
 clockrate 56000
 sdlc address C1
 sdlc xid C1 05D01501
 sdlc role primary
 fras map sdlc C1 serial 0 frame-relay 200 4 4
!
interface serial 0
 mtu 2500
 no ip address
 encapsulation frame-relay ietf
 keepalive 12
 frame-relay lmi-type ansi
 frame-relay map llc2 200
```

FRAS BNN Topology Example

FRAS BNN transports SNA traffic across different media through a Cisco router and then through a Frame Relay link to the host. SNA PU 2.0 and PU 2.1 devices may be attached to the remote router through Token Ring, SDLC, or Ethernet to access the Frame Relay network. The FRAS BNN topology is illustrated in Figure 193.

Figure 193 FRAS BNN Topology

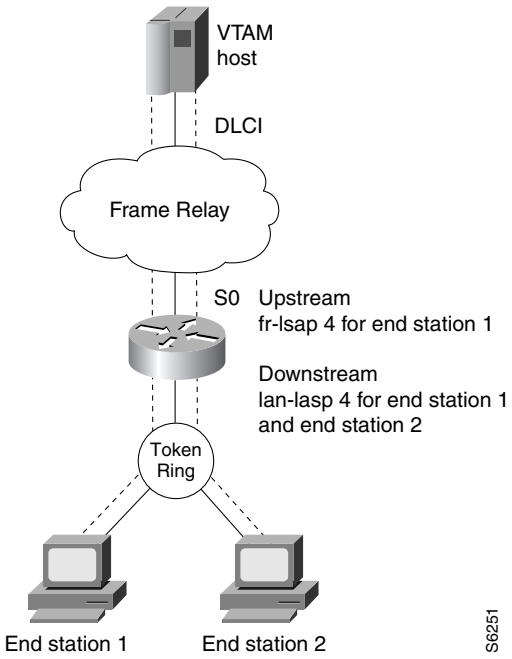


The original Frame Relay BNN feature transports traffic from multiple PUs over a single DLCI. This function is called SAP multiplexing. The router uses a unique SAP address (fr-lsap) for each downstream PU when communicating with the host. In this implementation, each end station's MAC/SAP address pair must be statically defined to the router. Consequently, the router must be reconfigured each time an end station is moved, added, or deleted. The configuration overhead for this implementation can be high.

The FRAS BNN feature, where the router "learns" the MAC/SAP information as it forwards packets to the host, offers several advantages over the original FRAS BNN implementation. The BNN enhancement alleviates the need to reconfigure the router when end stations are moved, added, or deleted. The configuration is simple: one map definition in the router is sufficient for multiple downstream devices. The router "learns" the addresses of the downstream devices in the normal course of communication (as shown in Figure 194).

Figure 194 illustrates the Frame Relay BNN configuration for both the original implementation and the enhanced implementation.

Figure 194 Frame Relay BNN Support

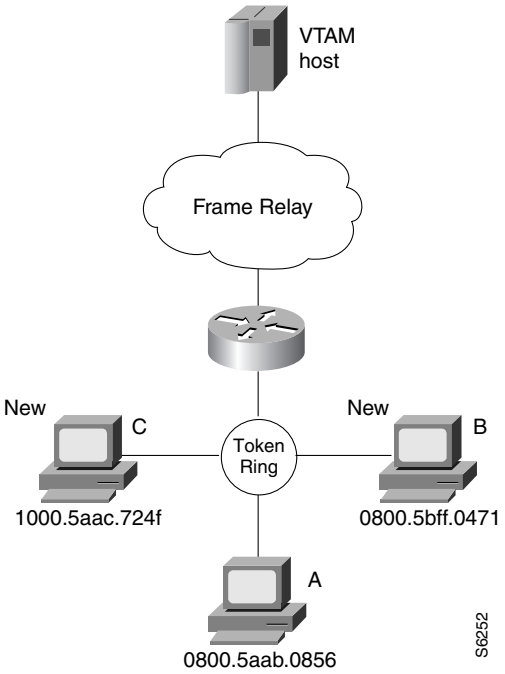


If the end station initiates the LLC session, the router acquires the Token Ring address and the SAP value of the end station from the incoming frame. Instead of mapping the end station's MAC/SAP address pair (as was done in the original FRAS BNN implementation), the destination MAC/SAP address pair of the incoming frame is mapped to the Frame Relay DLCI. If the destination SAP specified by the end station is equal to the lan-lsap address, the router associates the LLC (LAN) connection with the Frame Relay DLCI. The MAC address and the SAP address of the end station are no longer required in the router configuration. Thus, in the enhanced FRAS BNN implementation one configuration command achieves the same result for the end stations as did multiple configuration commands in the original FRAS BNN implementation.

FRAS BNN Example

The following configuration example enables the FRAS BNN feature. The topology is illustrated in Figure 195.

Figure 195 FRAS BNN Configuration




```
interface Serial0
 no ip address
 encapsulation frame-relay IETF
 frame-relay lmi-type ansi
 frame-relay map llc2 16
!
interface TokenRing0
 no ip address
 ring-speed 16
 fras map llc 0800.5aab.0856 04 04 Serial 0 frame-relay 16 04 04
 fras map llc 04 Serial 0 frame-relay dlci 16 04
```

Note In this configuration example, the second to last line describes the old configuration for workstation A. The last line describes the configuration for the new workstations B and C.

FRAS BAN Example

The following configuration shows FRAS BAN support for Token Ring and serial interfaces. You must specify the **source-bridge ring-group** global command before you configure the **fras ban** interface command. When Token Ring is configured, the **source-bridge** interface command includes the *local-ring*, *bridge-number*, and the *target-ring* values. The **source-bridge** command enables local source-route bridging on a Token Ring interface.

```
source-bridge ring-group 200
!
interface serial 0
 mtu 4000
 encapsulation frame-relay ietf
 frame-relay lmi-type ansi
 frame-relay map llc2  16
 frame-relay map llc2  17
 fras ban 120 1 200 4000.1000.2000 dlci 16 17
!
interface tokenring 0
 source-bridge 100 5 200
```

For SDLC connections, you must include SDLC configuration commands as follows:

```
!
interface Serial1
 description SDLC line PU2.0
 mtu 265
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 9600
 sdlc role primary
 sdlc vmac 4000.0000.0000
 sdlc address C2
 sdlc xid C2 05D01502
 sdlc partner 4000.0000.2345 C2
 sdlc address C8
 sdlc xid C8 05D01508
 sdlc partner 4000.0000.2345 C8
 sdlc address C9
 sdlc xid C9 05D01509
 sdlc partner 4000.0000.2345 C9
 fras ban frame-relay Serial0 4000.0000.2345 dlci 16
!
interface Serial2
 description SDLC line PU2.1
 no ip address
 encapsulation sdlc
 no keepalive
 clockrate 19200
 sdlc role prim-xid-poll
 sdlc vmac 2000.0000.0000
 sdlc address C6
 sdlc partner 1000.2000.3000 C6
 fras ban frame-relay serial0 1000.2000.3000 dlci 16
```

SRB over Frame Relay Example

Figure 196 illustrates the interoperability provided by SRB over Frame Relay. FRADs B and C forward frames from their locally attached Token Rings over the Frame Relay network using SRB.

Figure 196 FRAD Using SRB over Frame Relay to Connect to a Cisco Router

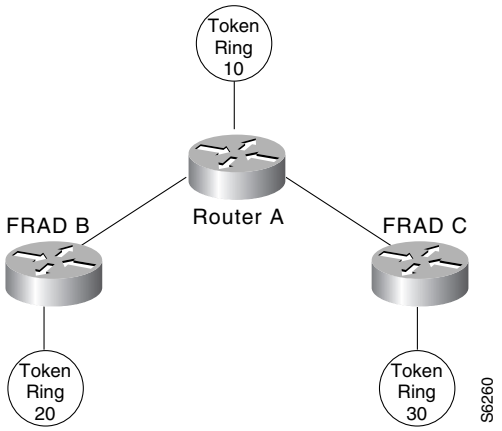


Figure 196 illustrates a network with the following characteristics:

- Virtual ring number of Router A = 100
- Virtual ring number of FRAD B = 200
- Virtual ring number of FRAD C = 300
- DLCI number for the partner's virtual ring (PVC) between Router A and FRAD B = 30
- DLCI number for PVC between Router A and FRAD C = 31

In this example we configure a new option, **conserve-ring**, on the **source-bridge** interface configuration command. When this option is configured, the SRB software does not add the ring number associated with the Frame Relay PVC to outbound explorer frames. This option is permitted for Frame Relay subinterfaces only.

The router configures the partner FRAD's virtual ring number as the ring number for the PVC.

This approach does not require a separate ring number per DLCI. The router configures the partner FRAD's virtual ring number as the ring number for the PVC.

FRAD B configures its virtual ring as 200 and the ring for the PVC as 100. FRAD C configures its virtual ring as 300 and the ring for the PVC as 100.

FRAS DLCI Backup over Serial Interface Example

The following example shows a configuration for FRAS DLCI backup over a serial interface:

```
interface serial0
 mtu 3000
 no ip address
 encapsulation frame-relay IETF
 bandwidth 56
 keepalive 11
 frame-relay map llc2 277
 frame-relay map llc2 278
 frame-relay lmi-type ansi
 fras ddr-backup interface serial1 188
!
interface serial1
 mtu 3000
 no ip address
 encapsulation frame-relay IETF
 no cdp enable
 frame-relay map llc2 188
 frame-relay lmi-type ansi
!
interface serial2
 no ip address
 encapsulation sdlc
 no keepalive
 clock rate 19200
 sdlc role prim-xid-poll
 sdlc address D6
 fras map sdlc D6 s0 frame-relay 277 8 4
!
interface tokenring0
 no ip address
 ring-speed 16
 fras map llc 0000.f63a.2f70 4 4 serial0 frame-relay 277 4 4
```

Router A

```
source-bridge ring-group 100
!
interface Serial1
 encapsulation frame-relay
!
interface Serial1.1 point-to-point
 frame-relay interface-dlci 30 ietf
 source-bridge 200 1 100 conserve-ring
 source-bridge spanning
!
interface Serial1.2 point-to-point
 frame-relay interface-dlci 31 ietf
 source-bridge 300 1 100 conserve-ring
 source-bridge spanning
!
interface TokenRing0
 source-bridge 500 1 100
```

FRAS Dial Backup over DLSw+ Example

The following examples show configurations for FRAS dial backup over DLSw+:

FRAS Dial Backup on a Subinterface

```
source-bridge ring-group 200
dlsw local-peer peer-id 10.8.8.8
dlsw remote-peer 0 tcp 10.8.8.7 dynamic
interface ethernet0
 ip address 10.8.8.8 255.255.255.0
!
interface serial0
 no ip address
 encapsulation frame-relay IETF
 frame-relay lmi-type ansi
!
interface Serial0.1 point-to-point
 description fras backup dlsw+ listening on dlci 16 configuration example
 no ip address
 frame-relay interface-dlci 16
 fras backup dlsw 4000.1000.2000 200 1000.5aed.1f53
!
interface TokenRing0
 no ip address
 ring-speed 16
 fras map llc 0000.f63a.2f50 4 4 Serial0.1 frame-relay 16 4 4
```

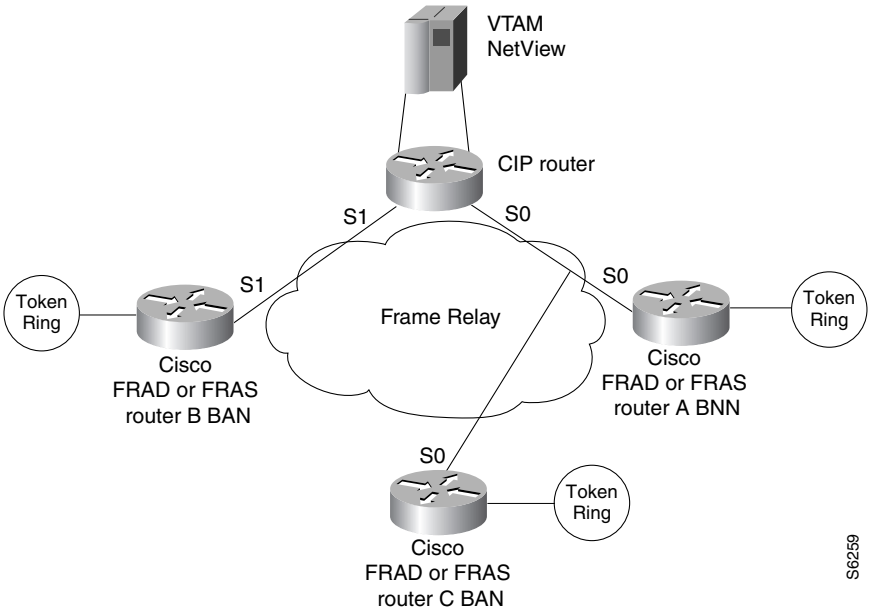
FRAS Dial Backup on a Main Interface

```
source-bridge ring-group 200
dlsw local-peer peer-id 10.8.8.8
dlsw remote-peer 0 tcp 10.8.8.7 dynamic
interface ethernet0
 ip address 10.8.8.8 255.255.255.0
!
interface serial0
 no ip address
 encapsulation frame-relay IETF
 frame-relay lmi-type ansi
frame-relay map llc2 16
fras backup dlsw 4000.1000.2000 200 1000.5aed.1f53
!
interface Serial1
 ip address 10.8.8.8
!
interface tokening0
 no ip address
 ring-speed 16
 fras map llc 0000.f63a.2f50 4 4 Serial0 frame-relay 16 4 4
```

Cisco FRAD or FRAS Router Configuration Examples

This section provides the following configuration examples (see Figure 197):

Figure 197 FRAS Host CIP Connection to VTAM



Cisco FRAD or FRAS Router A with BNN Configuration Example

```
interface Serial0
 encapsulation frame-relay IETF
 frame-relay map llc2 16
!
interface TokenRing0
 fras map llc 4001.2222.0000 4 4 Serial0 frame-relay 16 4 4
```

Cisco FRAD or FRAS Router B with BAN Configuration Example

```
source-bridge ring-group 200
!
interface Serial0
 encapsulation frame-relay IETF
 frame-relay map llc2 37
 fras ban 10 1 200 4000.3745.0000 dlci 37
!
interface TokenRing0
 source-bridge 20 1 200
```

Cisco FRAD or FRAS Router C with BAN Configuration Example

```
source-bridge ring-group 400
!
interface Serial0
 encapsulation frame-relay IETF
 frame-relay map llc2 46
 fras ban 50 1 400 4000.3745.0220 dlci 46 bni 4001.3745.1088
!
interface TokenRing0
 source-bridge 60 1 400
```

FRAS Host CIP Connection to VTAM Configuration Example

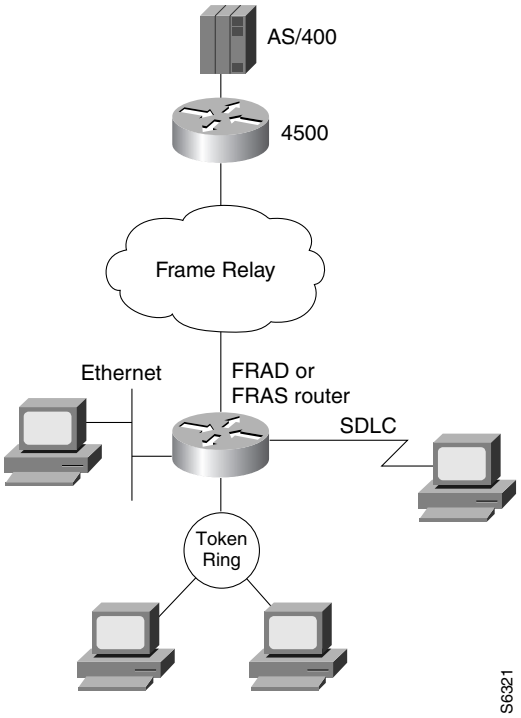
The following example shows the configuration for the network shown in Figure 198.

```
source-bridge ring-group 100
!
interface Serial0/1
 encapsulation frame-relay IETF
 frame-relay map llc2 16
 frame-relay map llc2 46
!
interface Serial0/2
 encapsulation frame-relay IETF
!
interface Serial0/2.37 point-to-point
 frame-relay interface-dlci 37
!
interface Channel4/0
 no keepalive
!
interface Channel4/1
 no keepalive
 lan TokenRing 0
  source-bridge 104 1 100
  adapter 0 4001.3745.1008
!
interface Virtual-TokenRing0
 source-bridge 47 1 100
 source-bridge spanning
 fras-host bnn Serial 0/1 fr-lsap 04 vmac 4005.3003.0000 hmac 4001.3745.1088
 fras-host ban Serial 0/1 hmac 4001.3745.1088 bni 4001.3745.1088
 fras-host ban Serial 0/2.37 hmac 4001.3745.1088
```

FRAS Host Ethernet Connection to AS/400 Configuration Example

The configuration example in this section is shown in Figure 198.

Figure 198 FRAS Host Ethernet Connection to AS/400




```
source-bridge ring-group 226
dlsw local-peer
dlsw bridge-group 1
!
interface Ethernet0
 bridge-group 1
!
interface Serial2
 encapsulation frame-relay IETF
 frame-relay map llc2 502
 frame-relay lmi-type ansi
!
interface Virtual-TokenRing0
 no ip address
 ring-speed 16
 source-bridge 1009 1 226
 fras-host dlsw-local-ack
 fras-host bnn Serial2 fr-lsap 04 vmac 4000.1226.0000 hmac 0800.5ae1.151d
```

Configuring NCIA Client/Server

This chapter describes native client interface architecture (NCIA) support for Systems Network Architecture (SNA) devices. NCIA server and the NCIA client/server model extends the scalability of NCIA I, the earlier NCIA implementation, by minimizing the number of central-site remote source-route bridging (RSRB) or data-link switching plus (DLSw+) peer connections required to support a large number of NCIA clients. For a complete description of the NCIA client/server commands mentioned in this chapter, refer to the "NCIA Server Configuration Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

Cisco's NCIA server feature implements RFC 2114, *Data Link Switch Client Access Protocol*. Using Cisco's RSRB technology, NCIA I encapsulates the Token Ring traffic inside IP datagrams passed over a TCP connection between a router and a client. A virtual ring is created to allow the router to interconnect any client. The virtual ring acts as a logical Token Ring in the router, so that all the Token Rings connected to the router are treated as if they are all on the same Token Ring. The virtual ring is

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

called a ring group. The ring group number is used just like a physical ring number and shows up in any route descriptors contained in packets being bridged. A ring group must be assigned a ring number that is unique throughout the network.

An NCIA I client acts as both an RSRB router and an end station. It must have a “fake” ring number and a “fake” bridge number so that it looks like an end station sitting on a real Token Ring. The fake ring and bridge numbers are visible to both the RSRB router and the NCIA client. The client must also have an LLC2 so that it can handle the LLC2 sessions.

NCIA Server

The NCIA Server feature extends the scalability of NCIA I, enhances its functionality, and provides support for both the installed base of RSRB routers and the growing number of DLSw+ routers. The NCIA Server feature includes the following enhancements:

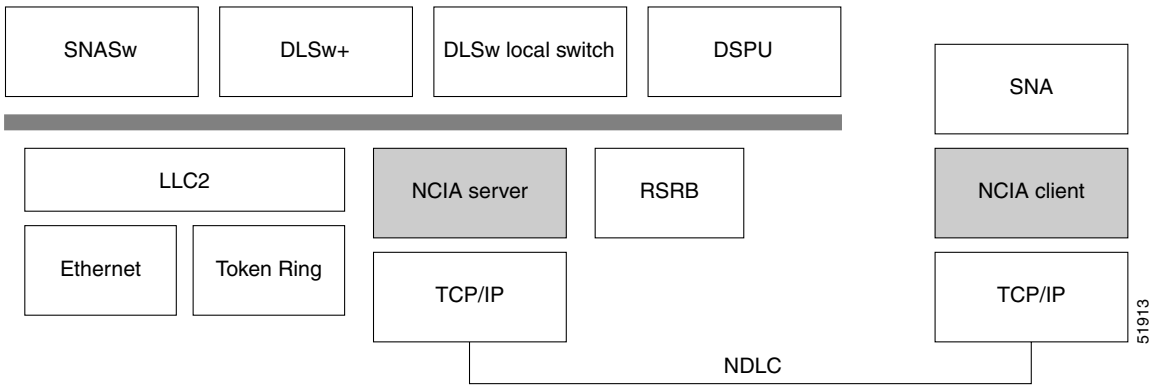
- You do not need to configure a ring number on the client.
- You do not need to configure each client on the router.
- The MAC address can be dynamically assigned by the NCIA server running on the router.
- SNA is directly on top of TCP/IP; LLC2 is no longer required at end station.
- A client is a true end station, not a router peer.
- The NCIA Server communicates with other components in router, such as RSRB, SNA Switching Services (SNASw), DLSw+, and DSPU.
- Supports both connect-in and connect-out.
- The NCIA client/server model is independent of the upstream implementation.
- It is an efficient protocol between client and server.

NCIA Client/Server Model

The NCIA Server feature uses a client/server model (Figure 199), where the NCIA server is a software module on a Cisco router and the NCIA client is a PC or workstation. The NCIA server performs two major functions:

- Establishes TCP to NCIA data-link Control (NDLC) sessions with clients for the purpose of sending and receiving data.
- Uses the Cisco link services interface (CLSI) to communicate with other software modules in the router, such as SNASw, DLSw+, and DSPU, and acts as the data intermediary between them and NCIA clients. The NCIA server’s role as an intermediary is transparent to the client.

Figure 199 NCIA Server Client/Server Model



NDLC is the protocol used between clients and servers. NDLC serves two purposes:

- Establishes the peer connection.
- Establishes the circuit between the client and the server.

The peer session must be established before an end-to-end circuit can be set up. During the set up period for the peer session, the MAC address representing a client is defined. The MAC address can be defined by the client or by the server when the client does not have a MAC address.

The NCIA Server feature supports connect-in and connect-out (from the server's perspective), but connect-out is not supported if the client station does not listen for the incoming connection. For a server to connect-out, clients must connect to the server first. After registering itself by providing its own MAC address, the client can then optionally disconnect from the server. When a server receives an explorer, and its destination MAC address is registered, an NCIA server will connect to that client if it is not connected. For NetBIOS explorers (addressed to functional address 0xC00000000080), the TCP session must remain up so that the server can broadcast the explorers to the client. If the TCP session is down, the server will not send the NetBIOS explorers to a client, even when the client is registered.

After the peer session has been established, the NDLC protocol establishes the circuit between the client and server. This circuit is used to transfer end-user data between the client and the server. Because the client and its target station are not on the same transport, they cannot form a direct, end-to-end circuit. Each client must form a circuit between the client and server, and the server must form another circuit between the server and the target station. The server links those two circuits to form an end-to-end circuit. The server acts as a mediator between the client and the target station so that packets can be transferred between them.

In the NCIA server only peer keepalive is maintained. There is no keepalive at circuit level.

The NCIA server acts as a data-link provider, like Token Ring or Ethernet, in the router. It uses CLSI to communicate with other software modules, just as other data-link providers do. The network administrator configures the router to communicate with specific modules. For data-link users, such as SNASw, DLSw+, and DSPU, the NCIA server can interface to them directly. For other data-link providers, the NCIA server must go through a DLSw+ local peer to communicate with them. The DLSw+ local peer passes packets back and forth among different data-link providers.

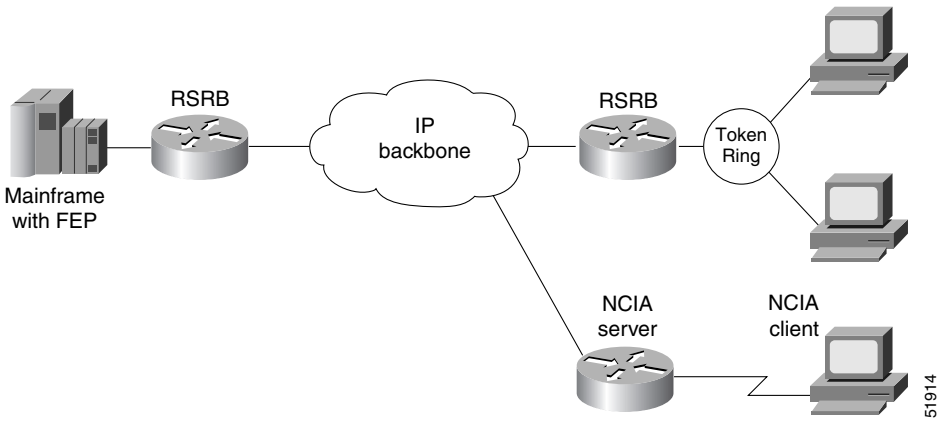
Advantages of the Client/Server Model

The client/server model used in the NCIA Server feature extends the scalability of NCIA. In addition, it provides support for both the installed base of RSRB routers and the growing number of DLSw+ routers.

Extended Scalability

The client/server model minimizes the number of central site RSRB or DLSw+ peer connections required to support a large network of NCIA clients (Figure 200). Rather than each client having a peer connection to a central site router, the clients attach to an IP backbone through an NCIA server that, in turn, has a single peer connection to a central site router. This scheme can greatly reduce the number of central site peer connections required. For example, in a network with 1000 clients and 10 NCIA servers, there would be only 10 central site peer connections. Note that there would still be 1000 LLC2 connections that must be locally acknowledged at the central site router, but this can easily be handled in a single central site router. When the number of LLC2 connections (or the number of clients) is in the tens of thousands, NCIA servers can take advantage of downstream PU concentration to minimize the number of LLC2 connections that must be supported by the central site routers.

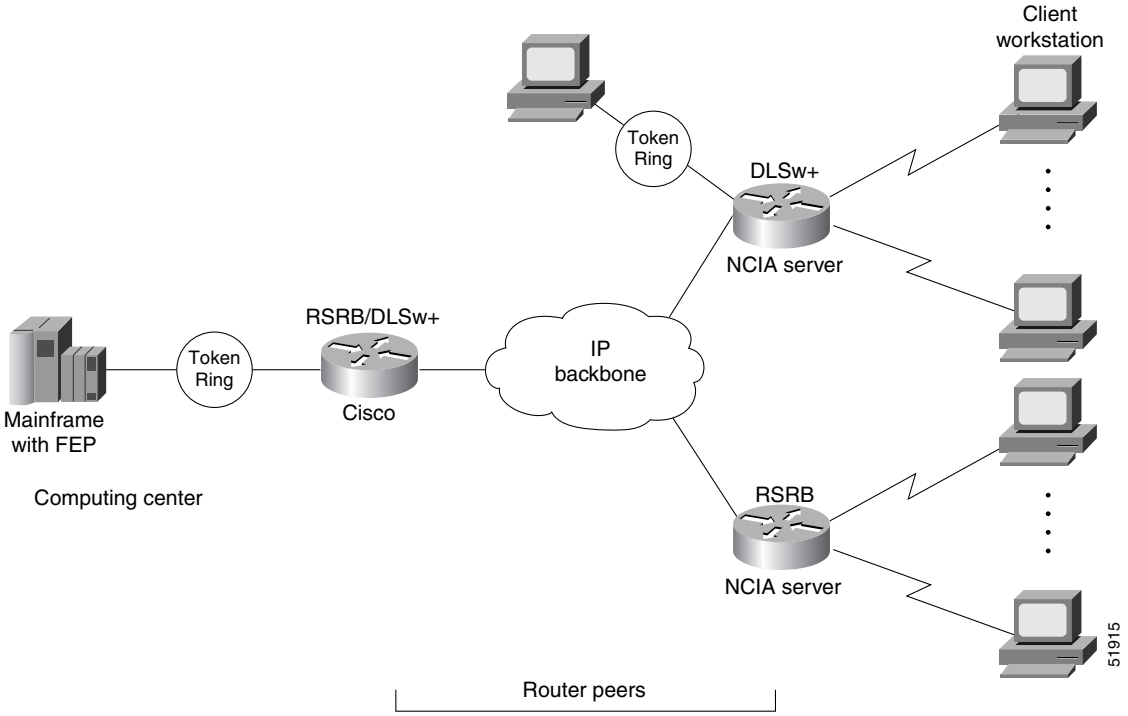
Figure 200 NCIA Server Provides Extended Scalability to Support Large Networks



Migration Support

Using a client/server model allows the NCIA Server feature to be independent of the upstream implementation, allowing it to be implemented in a network that is still using RSRB and in a DLSw+ network. It also greatly simplifies migration from RSRB to DLSw+, because it requires no changes at the client. A single NCIA server can support either approach (but not both). As Figure 201 illustrates, a central site router can support RSRB and DLSw+ concurrently, allowing a portion of the NCIA servers to communicate using RSRB and another portion to communicate using DLSw+.

Figure 201 NCIA Server Provides Independence from the Upstream Network Implementation



Configuring NCIA Server Session to Local Token Ring Using DLSw+ Local Switch

The network configuration shown in Figure 202 includes NCIA clients that connect to a front-end processor (FEP) on a Token Ring through a local router (the NCIA server). The virtual ring is used in conjunction with DLSw+ local switch. The routing information field (RIF) of each circuit is terminated on the virtual ring. Figure 203 shows a logical view of an NCIA server session using a DLSw+ local switch (connected to a local Token Ring). In addition to Token Ring, an NCIA server also supports Ethernet, Synchronous Data Link Control (SDLC) Protocol, and Qualified Logical Link Control (QLLC) network connections, and Channel Interface Processor (CIP) connections through a DLSw+ local switch. For more information on the different media types that a DLSw+ local switch supports, refer to the “Configuring DLSw+” chapter.

Figure 202 NCIA Server Session to Local Token Ring Using DLSw+ Local Switch

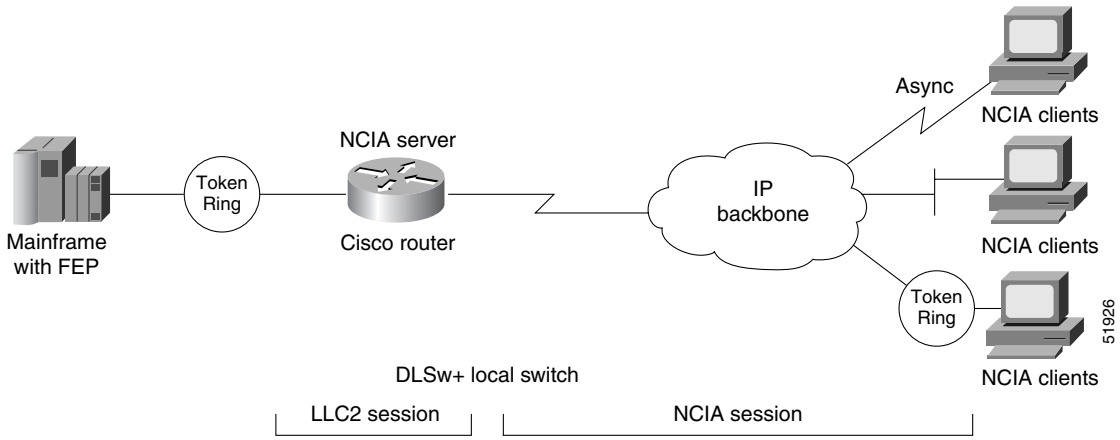
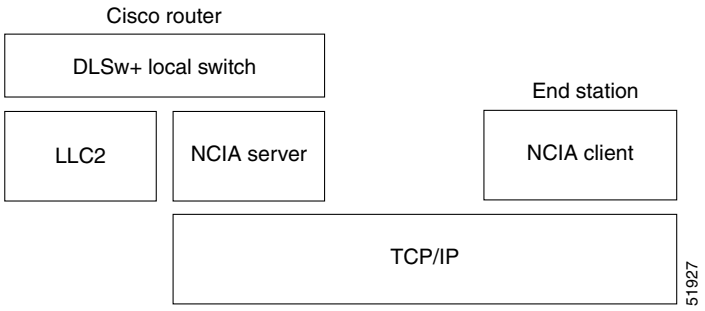


Figure 203 Logical View of NCIA Server Session to a Local Token Ring Using DLSw+ Local Switch



Configuration Task List

To configure an NCIA server session connected to a local Token Ring, perform the tasks in the following sections:

- Defining a Source-Bridge Ring Group for DLSw+, page 7
- Defining a DLSw+ Local Peer for the Router, page 7
- Configuring an NCIA Server on the Router, page 7

For a configuration example, see the "NCIA Server Session to Local Token Ring Using DLSw+ Local Switch Example" section on page 16.

Defining a Source-Bridge Ring Group for DLSw+

In DLSw+, the source-bridge ring group specifies the virtual ring that will appear to be the last ring in the RIF. This ring is transparent to the NCIA client. From the host's point of view, all NCIA clients look like stations sitting on the virtual ring. To define a source-bridge ring group for DLSw+, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.

Defining a DLSw+ Local Peer for the Router

Defining a DLSw+ local peer for a router enables a DLSw+ local switch. You specify all local DLSw+ parameters as part of the local peer definition. To define a local peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw local-peer [peer-id ip-address] [group group] [border] [cost cost] [lf size] [keepalive seconds] [passive] [promiscuous] [biu-segment]`	Defines the DLSw+ local peer.

Configuring an NCIA Server on the Router

Configuring an NCIA server on a router enables the router to perform two roles:

- Establish TCP/NDLC sessions with clients for the purpose of sending and receiving data.
- Use the standard interface (CLSI) to communicate with other software modules in the router, such as DLSw+, and DSPU, and act as the data intermediary between them and the clients of the NCIA server.

To configure an NCIA server, use the following command in global configuration mode:

Command	Purpose
`Router(config)# ncia server server-number server-ip-address server-virtual-mac-address virtual-mac-address virtual-mac-range [inbound-only] [keepalive seconds] [tcp_keepalive minutes]`	Configures the NCIA server.

Configuring NCIA Server Session with DLSw+

In the network configuration shown in Figure 204, the NCIA server uses DLSw+ to connect its clients to the FEP through a remote router. Figure 205 shows a logical view of the NCIA Server session with DLSw+.

Figure 204 NCIA Server Session with DLSw+

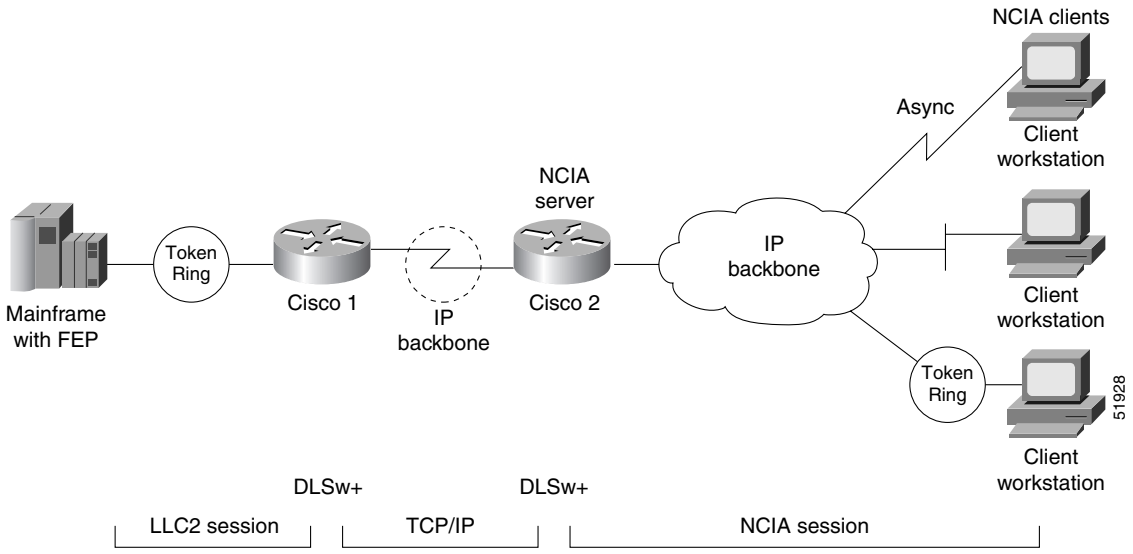
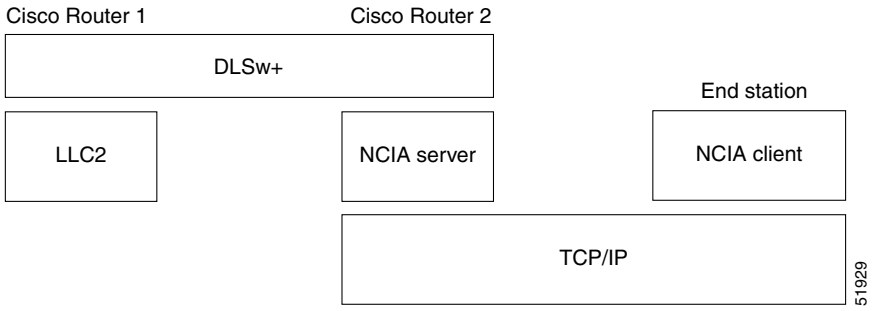


Figure 205 Logical View of NCIA Server with DLSw+



DLSw+ Configuration Task List

To configure an NCIA server session connected to a remote router using DLSw+, perform the tasks in the following sections:

- Defining a Source-Bridge Ring Group for DLSw+, page 9
- Defining a DLSw+ Local Peer for the Router, page 9
- Defining a DLSw+ Remote Peer, page 9
- Configuring an NCIA Server on the Local Router, page 10

For a configuration example, see the "NCIA Server Session with DLSw+ Example" section on page 17.

Defining a Source-Bridge Ring Group for DLSw+

The source-bridge ring can be shared between DLSw+ and SRB/RSRB. In DLSw+, the source-bridge ring group specifies the virtual ring that will appear to be the last ring in the RIF. Because RIFs are terminated at the router, there is no correlation between the ring-group number specified in DLSw+ peers. The numbers can be the same for management simplicity, but they do not have to be. To define a source-bridge ring group for DLSw+, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.

Defining a DLSw+ Local Peer for the Router

Defining a DLSw+ local peer for a router enables DLSw+. You specify all local DLSw+ parameters as part of the local peer definition. To define a local peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw local-peer [peer-id ip-address] [group group] [border] [cost cost] [lf size] [keepalive seconds] [passive] [promiscuous] [biu-segment]`	Defines the DLSw+ local peer.

Defining a DLSw+ Remote Peer

To configure TCP encapsulation on a remote peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw remote-peer list-number tcp ip-address [backup-peer ip-address] [bytes-netbios-out bytes-list-name] [cost cost] [dest-mac mac-address][dmac-output-list access-list-number] [dynamic] [host-netbios-out host-list-name] [inactivity minutes] [keepalive seconds] [lf size] [linger minutes] [lsap-output-list list] [no-llc minutes] [priority] [tcp-queue-max size] [timeout seconds]`	Defines a TCP encapsulation remote peer.

Configuring an NCIA Server on the Local Router

Configuring an NCIA server on the local router enables the router to perform two roles:

- Establish TCP/NDLC sessions with clients to send and receive data.
- Use the standard interface (CLSI) to communicate with other software modules in the router, such as DLSw+, and DSPU, and act as the data intermediary between them and the NCIA clients.

To configure an NCIA server, use the following command in global configuration mode:

Command	Purpose
`Router(config)# ncia server server-number server-ip-address server-virtual-mac-address virtual-mac-address virtual-mac-range [inbound-only] [keepalive seconds] [tcp_keepalive minutes]`	Configures the NCIA server.

Configuring NCIA Server Session with DSPU

In the network configuration shown in Figure 206, the NCIA server uses DSPU to connect its clients to the FEP through a remote router. Figure 207 shows a logical view of the NCIA server session with RSRB/DLSw+ and DSPU.

Figure 206 NCIA Server Session with DSPU

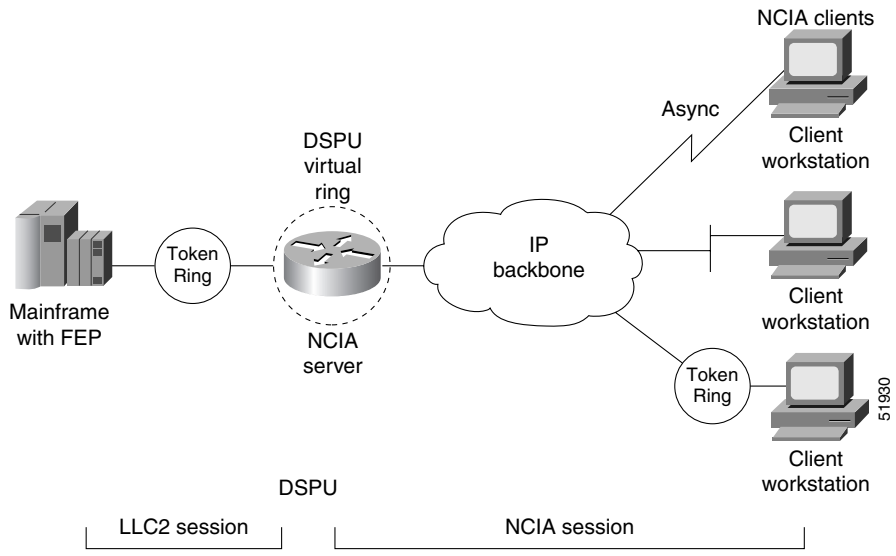
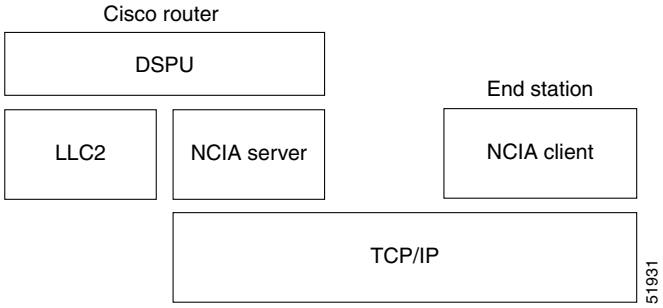


Figure 207 Logical View of NCIA Server with DSPU



DSPU Configuration Task List

To configure an NCIA server session connected to a remote router using DSPU, perform the tasks in the following sections:

- Defining a DSPU Upstream Host, page 11
- Explicitly Defining DSPU, page 11
- Defining Dedicated LU, page 12
- Configuring the NCIA Server as the Underlying Transport Mechanism, page 12

For a configuration example, see the "NCIA Server Session with DSPU Example" section on page 18.

Defining a DSPU Upstream Host

To define a DSPU host over Token Ring, Ethernet, Fiber Distributed Data Interface (FDDI), RSRB, or virtual data link control (VDLC) connections, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu host` *host-name* `xid-snd` *xid* `rmac` *remote-mac* `[rsap` *remote-sap*`] [lsap` *local-sap*`] [interface` *slot/port*`] [window` *window-size*`] [maxiframe` *max-iframe*`] [retries` *retry-count*`] [retry-timeout` *retry-timeout*`] [focalpoint]`	Defines a DSPU host over Token Ring, Ethernet, FDDI, RSRB, or VDLC connections.

Explicitly Defining DSPU

To explicitly define a DSPU over Token Ring, Ethernet, FDDI, RSRB, VDLC, or NCIA connections, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pu` *pu-name* `[rmac` *remote-mac*`] [rsap` *remote-sap*`] [lsap` *local-sap*`] [xid-rcv` *xid*`] [interface` *slot/port*`] [window` *window-size*`] [maxiframe` *max-iframe*`] [retries` *retry-count*`] [retry-timeout` *retry-timeout*`]`	Explicitly defines a DSPU over Token Ring, Ethernet, FDDI, RSRB, VDLC, or NCIA connections.

Defining Dedicated LU

To define a dedicated logical unit (LU) or a range of dedicated LUs for an upstream host and DSPU, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu lu lu-start [lu-end] {host host-name host-lu-start | pool pool-name} [pu pu-name]`	Defines a dedicated LU or a range of dedicated LUs for a DSPU.

Configuring the NCIA Server as the Underlying Transport Mechanism

To configure the NCIA server as the underlying transport mechanism, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu ncia [server-number]`	Configures the NCIA server as the underlying transport mechanism.

To enable a local service access point (SAP) on the NCIA server for use by DSPUs, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu ncia enable-pu [lsap local-sap]`	Enables local SAP for DSPUs.

Configuring NCIA Server Session with RSRB

The network configuration shown in Figure 208 includes NCIA clients that connect to a FEP on a Token Ring through a remote router. Figure 209 shows a logical view of the NCIA Server session with RSRB (to a remote Token Ring). Because DLSw+ is the latest technology provided by Cisco, Cisco does not encourage using the NCIA Server feature with RSRB. If the router on the host side is running DLSw+, then RSRB should not be used. Support for the NCIA Server feature with RSRB is provided to encourage RSRB users to migrate to DLSw+.

Figure 208 NCIA Server Session with RSRB

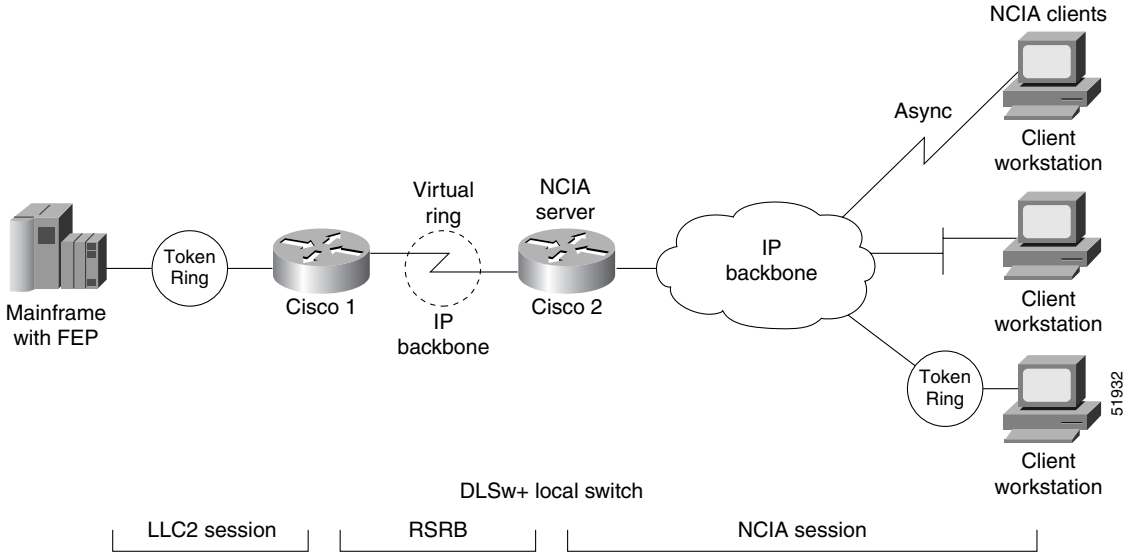
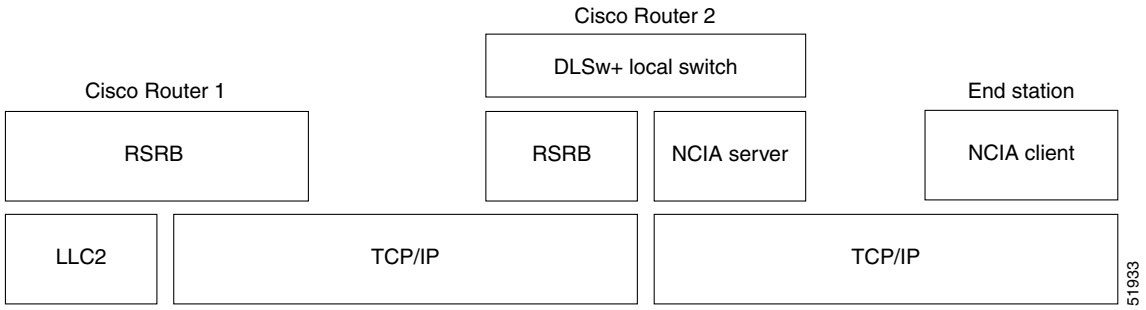


Figure 209 Logical View of NCIA Server Session with RSRB (Remote Token Ring)



RSRB Configuration Task List

To configure an NCIA server session connected to a remote Token Ring using RSRB, perform the tasks in the following sections:

- Defining a Source-Bridge Ring Group for DLSw+ and RSRB, page 14
- Identifying the Remote Peer (TCP Connection), page 14
- Defining a DLSw+ Local Peer for the Local Router, page 14
- Configuring an NCIA Server on the Router, page 15
- Configuring an RSRB Ring for the NCIA Server on the Local Router, page 15

For a configuration example, see the "NCIA Server Session with DLSw+ Example" section on page 17.

Defining a Source-Bridge Ring Group for DLSw+ and RSRB

The source-bridge virtual ring can be shared between DLSw+ and SRB/RSRB. In DLSw+, the source-bridge ring group specifies the virtual ring that will appear to be the last ring in the RIF. Because RIFs are terminated at the router, the ring group numbers specified in commands to set up DLSw+ peers can be different. The ring group numbers can be the same for management simplicity, but they do not have to be.

To define a source-bridge ring group for DLSw+, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines a ring group.

Identifying the Remote Peer (TCP Connection)

In our implementation, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router with which you want to exchange Token Ring traffic must be a member of this same ring group. For more information about defining a ring group, see the "Define a Ring Group in SRB Context" section of the "Configuring Source-Route Bridging" chapter of this document.

To identify the remote peers, use the following command in global configuration mode:

Command	Purpose
`Router(config)# source-bridge remote-peer ring-group tcp ip-address [lf size] [tcp-receive-window wsize] [local-ack] [priority]`	Identifies the IP address of a peer in the ring group with which to exchange source-bridge traffic using TCP.

Specify one **source-bridge remote-peer** command for each peer router that is part of the virtual ring. Also specify one **source-bridge remote-peer** command to identify the IP address of the local router.

NCIA server supports only RSRB pass-through mode. Local acknowledgment is not supported.

Defining a DLSw+ Local Peer for the Local Router

Defining a DLSw+ local peer for the local router enables DLSw+. You specify all local DLSw+ parameters as part of the local peer definition. To define a local peer, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dlsw local-peer [peer-id ip-address] [group group] [border] [cost cost] [lf size] [keepalive seconds] [passive] [promiscuous] [biu-segment]`	Defines the DLSw+ local peer.

Configuring an NCIA Server on the Router

Configuring an NCIA server on a router enables the router to perform two roles:

- Establish TCP/NDLC sessions with clients for the purpose of sending and receiving data.
- Use the standard interface (CLSI) to communicate with other software modules in the router, such as DLSw+, and DSPU, and to act as the data intermediary between them and the NCIA clients.

To configure an NCIA server, use the following command in global configuration mode:

Command	Purpose
`Router(config)# ncia server server-number server-ip-address server-virtual-mac-address virtual-mac-address virtual-mac-range [inbound-only] [keepalive seconds] [tcp_keepalive minutes]`	Configures the NCIA server.

Configuring an RSRB Ring for the NCIA Server on the Local Router

Configuring an RSRB ring to associate with the NCIA server on the local router provides the virtual ring that connects the DLSw ring within the local router and the target ring between the local router and the remote router.

To configure an RSRB ring for the NCIA server on the local router, use the following command in global configuration mode:

Command	Purpose
`Router(config)# ncia rsrb virtual-ring local-bridge local-ring ncia-bridge ncia-ring virtual-mac-address`	Defines the NCIA/RSRB interface.

Monitoring and Maintaining an NCIA Server Network

You can monitor and maintain the operation of an NCIA server network. To display information about the state of the NCIA server feature and perform maintenance tasks, use the following commands in EXEC mode:

Command	Purpose
`Router# clear ncia circuit [id-number]`	Drops an NCIA circuit.
`Router# clear ncia client [ip-address]`	Terminates an NCIA client connection.
`Router# clear ncia client registered [ip-address]`	Terminates the active connection to the specified client and release all control blocks of the registered client.
`Router# ncia start`	Restarts an NCIA server.
`Router# ncia stop`	Stops an NCIA server.
`Router# show ncia circuits [id-number]`	Shows the status of an NCIA circuit.
`Router# show ncia client [sap-list] [ip-address]`	Shows the status of the NCIA client.
`Router# show ncia server [server-number]`	Shows the status of the NCIA server.

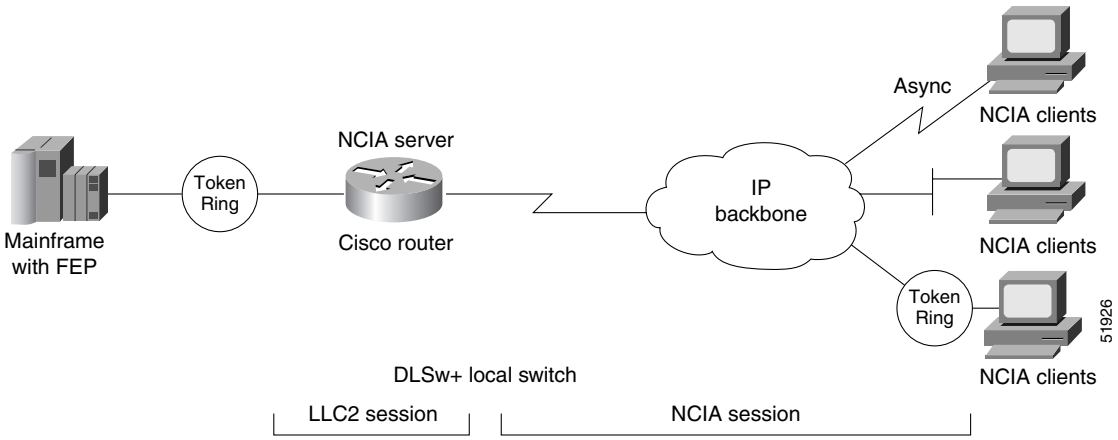
NCIA Server Configuration Examples

The following sections provide NCIA server configuration examples:

NCIA Server Session to Local Token Ring Using DLSw+ Local Switch Example

Figure 210 illustrates the use of DLSw+ local peer with an NCIA server session to a local Token Ring.

Figure 210 NCIA Server Session to Local Token Ring Using DLSw+ Local Switch



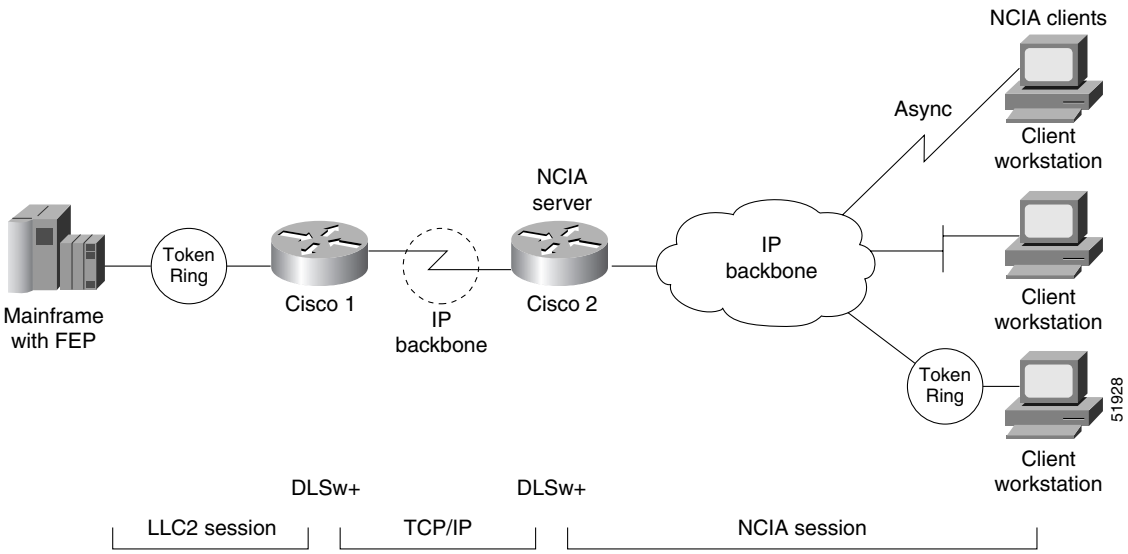
The following is a configuration for the network example shown in Figure 210:

```
source-bridge ring-group 44
dlsw local-peer
ncia server 1 10.2.20.4 4000.3174.0001 4000.0000.0001 128
!
interface token ring 0
 ring-speed 16
 source-bridge 21 3 44
```

NCIA Server Session with DLSw+ Example

Figure 211 illustrates the use of DLSw+ with an NCIA server session.

Figure 211 NCIA Server Session with DLSw+



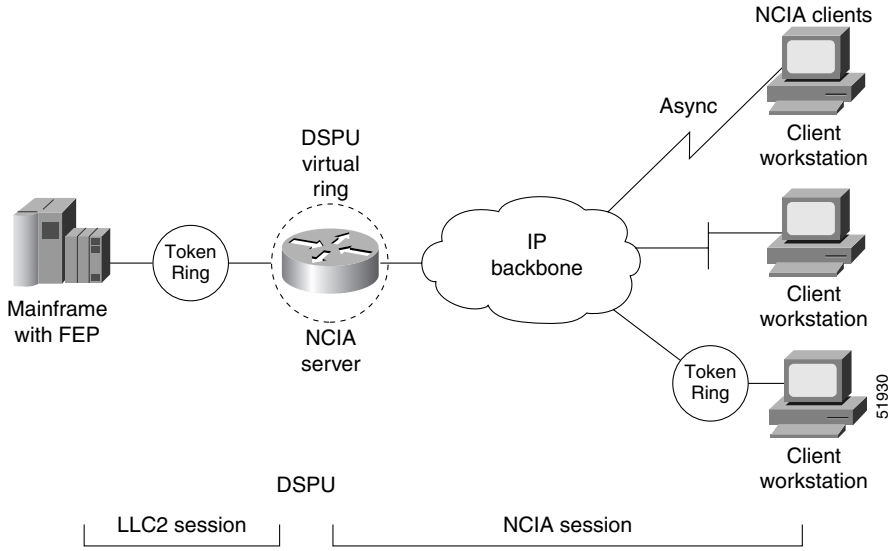
The following is a configuration for the network example shown in Figure 211:

```
source-bridge ring-group 44
dlsw local-peer peer-id 10.2.20.4
dlsw remote-peer 0 tcp 10.2.20.3
ncia server 1 10.2.20.4 4000.3174.0001 4000.0000.0001 128
```

NCIA Server Session with DSPU Example

Figure 212 illustrates an NCIA server session with RSRB/DLSw+ and DSPU.

Figure 212 NCIA Server Session with RSRB/DLSw+ and DSPU



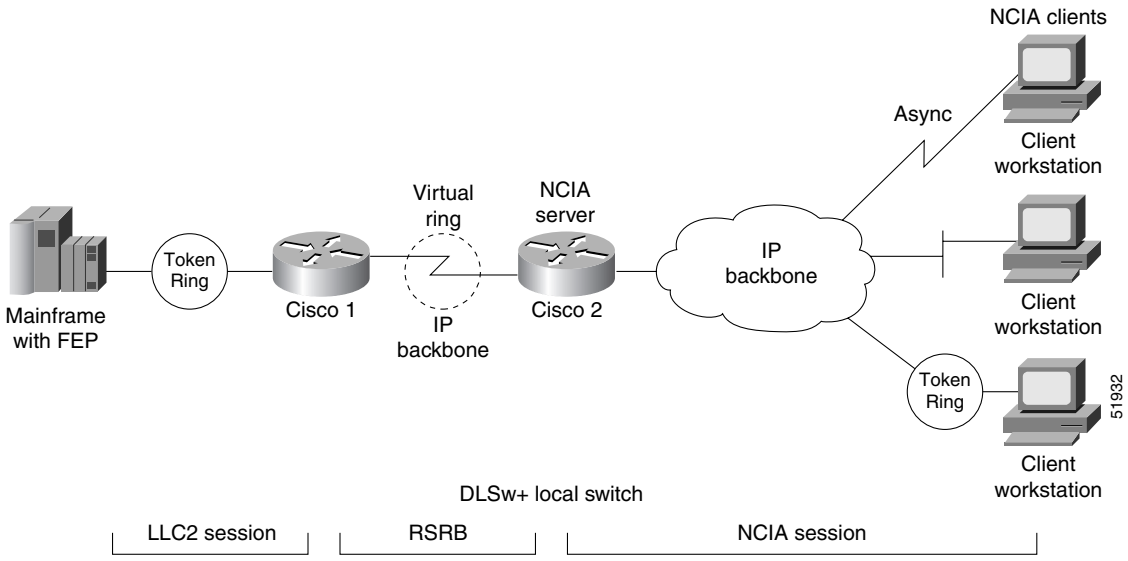
The following is a configuration for the network example shown in Figure 212:

```
ncia server 1 10.2.20.4 4000.3745.0001 4000.0000.0001 128
!
dspu ncia 1
dspu ncia enable-pu lsap 8
!
dspu host HOST-9370 xid-snd 11100001 rmac 4000.1060.1000 rsap 4 lsap 4
!
dspu pu CISCOPU-A xid-rcv 01700001
dspu lu 2 6 host HOST-9370 2
!
interface TokenRing 0
 ring-speed 16
 llc2 xid-retry-time 0
 dspu enable-host lsap 4
 dspu start HOST-9370
```

NCIA Server Session with RSRB Example

Figure 213 illustrates the use of RSRB with an NCIA server session.

Figure 213 NCIA Server Session with RSRB



The following is a configuration for router Cisco 2 for the network example shown in Figure 213:

```
source-bridge ring-group 44
source-bridge ring-group 22
source-bridge remote-peer 22 tcp 10.2.20.3
source-bridge remote-peer 22 tcp 10.2.20.4
dlsw local-peer
ncia server 1 10.2.20.4 4000.3174.0001 4000.0000.0001 128
ncia rsrb 22 2 33 4 44 1111.1111.2222
```

Configuring the Airline Product Set

This chapter describes how to configure the Airline Product Set (ALPS). For a complete description of the ALPS commands in this chapter, refer to the "Airline Product Set Configuration Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

ALPS Overview

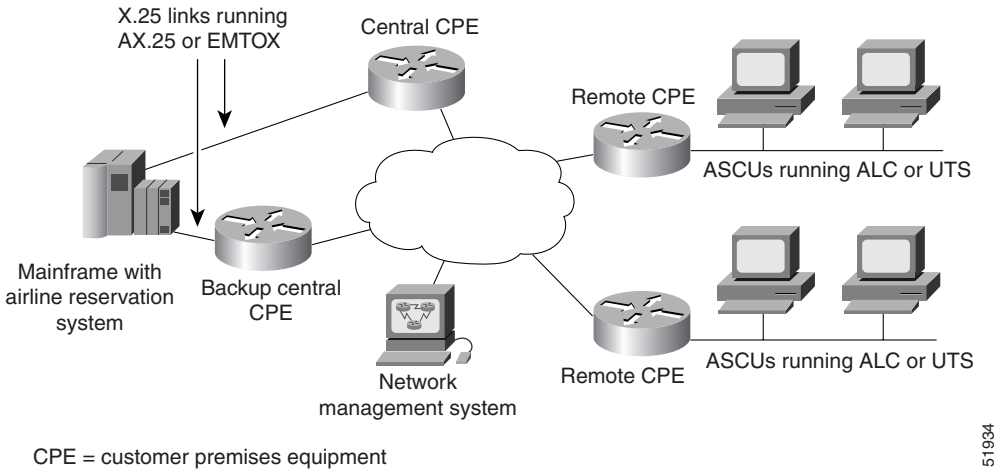
ALPS is a tunneling mechanism that transports airline protocol data across a Cisco router-based TCP/IP network to a mainframe. This feature provides connectivity between agent set control units (ASCUs) and a mainframe host that runs the airline reservation system.

The ALPS feature was released in three phases. The first two phases of ALPS enabled the network migration to TCP/IP without requiring any changes in the hardware or software of the endstations (ASCUs and mainframes). ALPS phase I and II utilized a new protocol, ALPS Tunneling Protocol (ATP), to tunnel airline protocol traffic (P1024B Airline Control [ALC] or P1024C Universal Terminal Support [UTS] data) through the TCP/IP network between peer Cisco routers. ALPS phase I provided support for the ALC protocol and the transport of the data from the ASCUs to a reservations system on an IBM mainframe. ALPS phase II provided support for the UTS protocol and the transport of the data from the ASCUs to a reservations system on a Unisys host system.

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

Figure 214 shows a basic ALPS topology with ALC, UTS, AX.25 and Exchange of Mixed Traffic over X.25 SVCs (EMTOX) protocols. Three major components provide the end-to-end transportation of airline protocol traffic across the network: the P1024B ALC or P1024C UTS protocol, the TCP-based transport protocol, and the AX.25/EMTOX access to the mainframe.

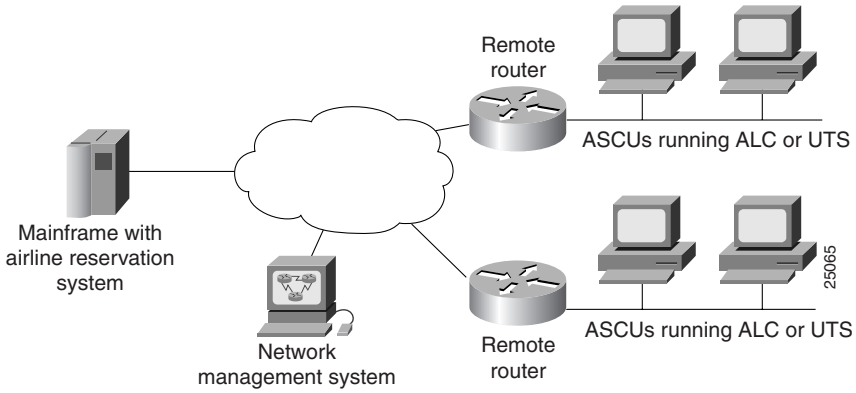
Figure 214 ALPS with ALC and UTS Architecture



ALPS phase III provides support for Mapping of Airline Traffic over Internet Protocol (MATIP). MATIP is an industry standard protocol for transporting airline protocol traffic across a TCP/IP network. This enhancement enables the end-to-end delivery of ALC and UTS data streams between a Cisco router and the mainframe using TCP/IP. ALPS with MATIP removes the X.25 (AX.25 or EMTOX) requirements for communication with the host reservation system by enabling TCP/IP communication between the router and the airline host reservation system.

Figure 215 shows the basic ALPS topology and the MATIP architecture implemented in Phase III. Three major components provide the end-to-end transportation of airline protocol traffic across the network: the P1024B ALC or P1024C UTS protocol, the TCP/IP-based MATIP protocol conversion, and the TCP/IP access to the mainframe.

Figure 215 ALPS with MATIP Architecture



In Cisco IOS Release 12.1(2)T and later, ALPS supports service messages additions and extensions to the ALPS P1024B ALC protocol support. The additions include customized options to configure the format, address, and sending of service messages. The ALPS ALC support is extended to be more scalable. The ALPS ASCU debug support is extended to include trace capability for the six-bit International Programmable Airline Reservation System (IPARS) format.

The Cisco ALPS feature provides the following benefits:

- Provides an end-to-end solution for airlines and central reservation systems.
- Allows airlines to replace their existing hardware and software with Cisco routers because the ALPS feature is integrated in the Cisco IOS software. For customers who already use Cisco routers, this feature allows them to consolidate networking overhead and functionality.
- Enables the end-to-end delivery of ALC and UTS data between a remote router or gateway and the mainframe using TCP/IP encapsulation.
- Eliminates network overhead for error detection and transmission logic associated with X.25 links.
- Replaces IBM front-end processors (FEPs) with Channel Interface Processors (CIPs).
- Eliminates the use of dedicated, leased, slow-speed ALC and UTS serial lines and migrates the reservation system networks to a modern networking paradigm.Once the mainframe reservation system is enabled to use TCP/IP, new applications can be written for PCs or network computers (NCs).
- Supports standards-based MATIP protocol for transporting data across the TCP/IP network.

In Cisco IOS Release 12.1(2)T and later, ALPS includes the following debug, ALC, and service message enhancements.

Debug Enhancement

The ALPS ASCU debug support additions provide new capabilities that enable you to display **debug alps ascu** command trace output in IPARS format.

ALC Enhancements

The ALPS ALC protocol stack includes the following extensions:

- Automatic ASCU reset
- T1 timer range increase
- Modification of the accepted ASCU IA value list

Service Message Enhancements

The additions to the ALPS service messages provide new capabilities that enable you to:

- Specify sita or apollo service message format
- Disable the forwarding of service messages for ALPS circuit status changes
- Specify where to retrieve the terminal address for dropped-data service messages
- Disable specific service messages
- Configure service message text with an increased character length

In Cisco IOS Release 12.1(3)T and later, ALPS includes the following ALC enhancement.

ALC Enhancement

The ALPS ALC protocol stack includes the following extensions:

- Nonpolled ALC ASCU support

The ALPS feature supports only type A conversational protocol traffic. The ALPS feature does not support MATIP type A host-to-host protocol traffic and MATIP type B messaging protocol traffic.

Remote routers must have the Cirrus Logic CD2430 chipset on a synchronous serial interface module to connect to the ALC or UTS ASCUs. The CD2430 chipset is supported on the following router platforms:

- Cisco 2520, 2521, 2522, and 2523
- Cisco 2600 series
- Cisco 3600 series
- Cisco 4500
- Cisco 4700

Note The Cisco 4500 and Cisco 4700 platforms must have a high-density, low-speed serial card installed. Sixteen low-speed ports are available for performing the remote router functions.

The ALPS feature supports the following standards, MIBs and RFCs:

Standards

- *P1024B Communication Control Protocol Specification*, Societe Internationale de Telecommunications Aeronautiques
- *P1024C Communication Control Protocol Specification*, Societe Internationale de Telecommunications Aeronautiques
- *MATIP Implementation Guide*, Societe Internationale de Telecommunications Aeronautiques

MIBs

The ALPS feature supports the CISCO-ALPS-MIB and the following MIB enhancements:

- Extensions to the alpsIfP1024Table
- Extension to the alpsAscuTable
- Addition of Simple Network Management Protocol (SNMP) notifications for ALPS circuit open request failure and ALPS circuit open request with a partial rejection

For descriptions of supported MIBs and how to use them, see the Cisco MIB website on Cisco.com.

RFCs

- RFC 2351, *Mapping of Airline Reservation, Ticketing, and Messaging Traffic over IP*, May 1998

ALPS Configuration Task List

See the following sections for configuration tasks for the ALPS feature. Each task in the list indicates if the task is optional or required. The tasks in the "Configuring the Remote Routers" section on page 6 are the only required tasks for ALPS with MATIP.

For a complete description of the ALPS commands in this feature module, refer to the "Airline Product Set Configuration Commands" chapter in the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2). To locate documentation of other commands, use the command reference master index or search online.

- Customizing the Alarm Notifications, page 12 (Optional)
- Updating a Circuit, page 12 (Optional)
- Verifying ALPS, page 13 (Required)

See the "ALPS Configuration Examples" section on page 14 for more information.

Configuring the Remote Routers

Note To configure ALPS with MATIP, you must perform only the following tasks. The tasks also apply to EMTOX and AX.25, but are not required.

Perform the tasks in the following sections to configure the ALPS feature on the remote routers:

- Specifying the ALPS Local Peer IP Address, page 6
- Specifying the ALPS Remote Peer IP Address, page 6
- Specifying the ALPS Circuit, page 7
- Specifying Each ASCU, page 8

Specifying the ALPS Local Peer IP Address

You must identify an IP address as an ALPS local peer on the remote router. Only one ALPS local peer is permitted on a router.

To specify the ALPS local peer IP address, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# alps local-peer` *ipaddress* `[promiscuous]`	Specifies an IP address to use as the ALPS local peer on the remote router.
Step 2	`Router(config)# alps keepalive [interval` *time*`] [retry` *count*`]`	Enables TCP keepalives for ALPS TCP peer connections.

Specifying the ALPS Remote Peer IP Address

You must specify a partner IP address (remote peer) on the remote router. The peer connection may be permanent or dynamic (established on demand). You can configure an ATP connection to be permanent or dynamic by configuring the optional **dynamic** keyword.

Note MATIP sessions are dynamic, whether or not the **dynamic** keyword is configured. To simulate a permanent connection in MATIP, configure the **dynamic** keyword with an *inact-timer* value of zero.

To specify the partner IP address for one or more TCP peer connections to the configured IP address, use the following command in global configuration mode:

Command	Purpose
`Router(config)# alps remote-peer ip-addr [protocol {atp | matip-a}] [status-interval interval] [status-retry retries] [dynamic [inact-timer] [no-circuit no-circ-timer]] [tcp-qlen [num]]`	Specifies the partner IP address. If you select the ATP protocol, you must configure the data center routers.

Specifying the ALPS Circuit

An ALPS circuit is a communication path across a TCP connection for one or more ASCUs. The ALPS circuit must have a configured association with an ALPS remote peer to establish a connection to the host. Additionally, an ALPS circuit configuration may specify a different remote peer as a backup peer to the host. Each MATIP circuit maps to a single TCP connection. For ATP, ALPS circuits can be multiplexed across to a single TCP connection.

To specify an ALPS circuit, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# alps circuit name`	Specifies an ALPS circuit at the remote router and enters ALPS circuit submode.
Step 2	`Router(config-alps-circ)# alps primary-peer ip-addr [backup-peer ip-addr]`	Specifies the primary TCP peer and an optional backup peer for this ALPS circuit.
Step 3	`Router(config-alps-circ)# alps local-hld loc-hld remote-hld rem-hld`	Specifies the local high-level designator (HLD) for this ALPS circuit. The **remote-hld** keyword is not applicable for ALPS with MATIP. The *loc-hld* is the hld of the device that is being replaced. The *rem-hld* is the hld of the host mainframe.
Step 4	`Router(config-alps-circ)# alps hostlink number {ax25 lcn | emtox x121-addr} [winout val1] [winin val2] [ops val3] [ips val4]`	Specifies information required to establish an X.25 virtual circuit at the central CPE.
Step 5	`Router(config-alps-circ)# alps connection-type permanent [retry-timer]`	(Optional) Specifies that this circuit should be established when the circuit is enabled.
Step 6	`Router(config-alps-circ)# alps lifetime-timer timer`	(Optional) Specifies how long messages can be queued in the ALPS circuit queue.
Step 7	`Router(config-alps-circ)# alps service-msg-interval seconds`	(Optional) Specifies the interval between the transmission of a service message to an ASCU and the transmission of a PLEASE RETRY message. The PLEASE RETRY message is transmitted only to ASCUs that use circuits with a dynamic connection type.
Step 8	`Router(config-alps-circ)# alps service-msg-list list`	(Optional) Defines the service message list to be used for this circuit.
Step 9	`Router(config-alps-circ)# alps matip-close-delay time`	(Optional) Specifies the interval between the closing and reopening of the MATIP circuit connection.

	Command	Purpose
Step 10	`Router(config-alps-circ)# alps idle-timer timer`	(Optional) Specifies (for dynamic circuits) the length of time that can elapse before an idle circuit is disabled.
Step 11	`Router(config-alps-circ)# alps mpx {group | single} hdr {a1a2 | none}`	(Optional) Specifies the multiplexing and the ASCU identification header for this circuit.
Step 12	`Router(config-alps-circ)# alps enable-circuit`	Enables the circuit.

Specifying Each ASCU

You must configure each ASCU within the context of the serial interface configuration. You must configure ASCU addressing information and association with an ASCU. You can configure the timers, maximum frame sizes, retry values, and polling mode optional configuration parameters for each ASCU. Appropriate default parameters are used for unspecified parameters. Once you configure the first ASCU, you can configure additional ASCUs using only Steps 8 through 14.

To specify an ASCU, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type number`	Configures an interface and enters interface configuration mode.
Step 2	`Router(config-if)# encapsulation [alc | uts]`	Specifies the protocol to be used on the serial interface.
Step 3	`Router(config-if)# alps t1 delay`	(Optional) Specifies the timeout delay between the transmission of an ALC poll message and the receipt of the first character of the poll message response.
Step 4	`Router(config-if)# alps t2 delay`	(Optional—ALC only) Specifies the timeout delay between receipt of the first character of the response to a poll message and the receipt of a Go Ahead message. Applies to ALC, only.
Step 5	`Router(config-if)# alps n1 errors`	(Optional) Specifies the threshold of consecutive errors logged before an ASCU is declared down.
Step 6	`Router(config-if)# alps n2 polls`	(Optional) Specifies the number of polls that must be correctly replied to before an ASCU is declared up.
Step 7	`Router(config-if)# alps n3 value`	(Optional—UTS only) Specifies the maximum number of retransmissions of an unacknowledged output data message to an ASCU. Applies only to UTS.
Step 8	`Router(config-if)# alps servlim polls`	(Optional) Specifies the number of cycles of the active poll list to execute before polling the next ASCU on the inactive poll list.
Step 9	`Router(config-if)# transmitter-delay delay`	Specifies number of padding characters added to the end of the frame (minimum dead-time after transmitting a packet).

	Command	Purpose
Step 10	`Router(config-if)# half-duplex`	Specifies half-duplex mode on a serial interface. This command specifies whether hardware flow control (constant or switched Request to Send [RTS]) is to be used between a DTE and DCE device. • If half-duplex is specified for a DTE, the DTE raises RTS and waits for the DCE to raise Clear to Send (CTS) before sending. • If half-duplex is specified for a DCE, the DCE waits for the DTE to raise RTS, then the DCE raises CTS to allow the DTE to send. • If full-duplex is specified, RTS is assumed and CTS is not monitored. **Note** ALPS supports the serial interface commands that are available if half-duplex mode is specified. This support applies to an interface that is configured as data circuit-terminating equipment (DCE) and data terminal equipment (DTE).
Step 11	`Router(config-if)# alps poll-pause msec`	(Optional) Specifies the minimum interval, in milliseconds, between initiations of the polling cycle.
Step 12	`Router(config-if)# alps service-msg data-drop {msg-term | config-term}`	(Optional) Specifies where to retrieve the terminal address to use when a service message is sent to an ASCU as a result of a dropped data message from a terminal.
Step 13	`Router(config-if)# alps service-msg format {sita | apollo}`	(Optional) Specifies the protocol format of service messages sent from the router to an ASCU.
Step 14	`Router(config-if)# alps service-msg status-change`	(Optional) Specifies that service messages for ALPS circuit status changes will be sent to ASCUs on the serial interface.
Step 15	`Router(config-if)# alps ascu id`	Specifies a physical ASCU identity (the ASCU interchange address value for ALC) and enters ALPS ASCU submode.
Step 16	`Router(config-alps-ascu)# alps default-circuit name`	Specifies the ALPS circuit that this ASCU uses.
Step 17	`Router(config-alps-ascu)# alps a1-map a1-value a2-map a2-value`	Specifies the A1 and A2 logical ASCU identification information.
Step 18	`Router(config-alps-ascu)# alps retry-option [resend | reenter]`	(Optional) Specifies the retry option when an ALC message with a bad cyclic check character (CCC) is received.
Step 19	`Router(config-alps-ascu)# alps max-msg-length value`	(Optional) Specifies maximum input message length.

	Command	Purpose
Step 20	`Router(config-alps-ascu)# alps error-display number1 number2`	(Optional) Specifies where error messages are displayed: • For P1024B ALC, the *number1* argument specifies the terminal address (TA) where these service messages are sent and the *number2* argument specifies the screen line number where service messages are displayed. • For P1024C UTS, the *number1* argument specifies the screen line number where service messages are displayed and *number2* argument specifies the column number where service messages are displayed.
Step 21	`Router(config-alps-ascu)# alps auto-reset`	(Optional) Automatically resets non-responsive ALC ASCUs in the DOWN state.
Step 22	`Router(config-alps-ascu)# alps alias alias-interchange-address`	(Optional) Specifies that an ALC ASCU is to operate in nonpolling mode and specifies the parent ASCU interchange address to which this ASCU is aliased.
Step 23	`Router(config-alps-ascu)# alps enable-ascu`	Polls the ASCU.

Configuring the Data Center Router

Note These tasks apply to EMTOX and AX.25, only.

Perform the tasks in the following sections to configure the ALPS feature on the data center router:

- Specifying the ALPS Host Local Peer Address, page 10
- Specifying AX.25, page 11
- Specifying EMTOX, page 11

Specifying the ALPS Host Local Peer Address

You must identify an IP address to use as the ALPS local peer IP address. Only one ALPS host local peer is permitted on a router. The promiscuous option, which allows any remote router to connect, is recommended at the central CPE.

To specify the ALPS host local peer address, use the following command in global configuration mode:

Command	Purpose
`Router(config)# alps local-peer ip-address [promiscuous]`	Specifies the IP address of the local peer.

Specifying AX.25

To enable AX.25 on an X.25 interface, the ALPS host HLD and hostlink number must be configured and AX.25 must be specified on an X.25 serial interface. At circuit-establishment time, the remote router forwards the host HLD, the logical channel number (LCN), and the hostlink number for the permanent virtual circuit (PVC), to be used for the ASCU group.

To configure AX.25 on an X.25 interface, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type number`	Configures an interface and enters interface configuration mode.
Step 2	`Router(config-if)# encapsulation x25`	Specifies a serial interface as an X.25 device.
Step 3	`Router(config-if)# alps host-hld hld host-link num {{ax25 [damp-tmr value]} | {emtox x.121 [pseudo-conv]}} [life-tmr value]`	Enables ALPS on the X.25 interface.

Specifying EMTOX

To enable EMTOX on an X.25 interface, the host HLD and the hostlink number must be configured and EMTOX must be specified on an X.25 serial interface. At circuit-establishment time, the remote router forwards the X.121 address to be used as the calling address in the X.25 call and the host HLD and the hostlink number. If the host performs a call out, a correlation between the X.121 called address and a remote router peer IP address must be configured.

To configure EMTOX on an X.25 interface, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type number`	Configures an interface and enters interface configuration mode.
Step 2	`Router(config-if)# encapsulation x25`	Specifies a serial interface as an X.25 device.
Step 3	`Router(config-if)# alps host-hld hld host-link num {{ax25 [damp-tmr value]} | {emtox x.121 [pseudo-conv]}} [life-tmr value]`	Enables ALPS on the X.25 interface.
Step 4	`Router(config-if)# alps translate x.121-addr ip-addr`	Maps an X.121 address to an IP address on a remote peer.

Customizing the Service Messages

You can customize the contents of the service messages and service message list. To specify the service message number and the content of the message, use the following command in global configuration mode:

Command	Purpose
`Router(config)# alps service-msg-list list number number msg`	Specifies service message numbers and content.

Note The default service message is used if no service message list number is specified. If you configure a particular service message on a list, the default service message still is used for the rest of the messages on that list.

Note Once the **alps service-msg-list number** command has been configured, you can define the service message list to be used on the circuit by configuring the **alps service-msg-list** command.

Note You can configure the handling of service messages using the **alps service-msg data-drop**, **alps service-msg format**, and **alps service-msg status-change** interface configuration-level commands.

Table 8 shows the default service message text strings:

Table 8 Service Message Default Text Strings

Message Number	Event	Text String
1	ALPS circuit to host is opened.	CONNECTION UP
2	X.25 virtual circuit at the host has been cleared.	DISC BY THE HOST
3	X.25 interface at the host is down.	HOST ISOLATED
4	No response from the host router when trying to establish a connection.	NETWORK PROBLEM
5	Connection to host was disconnected because of inactivity.	READY TO CONNECT
6	Network is congested.	CONGESTION
7	Network congestion has cleared.	PLEASE PROCEED
8	Network operator has disabled the path to the host.	DISC BY NET OPERAT

Customizing the Alarm Notifications

You can enable and customize alarms (error messages) and SNMP traps. To enable and customize alarms for the ALPS ASCUs, circuits, or peers, use the following commands in global configuration mode:

Command	Purpose
`Router(config)# alps enable-alarms ascu [interface id]`	Enables alarms for the ALPS ASCUs.
`Router(config)# alps enable-alarms circuit [name]`	Enables alarms for the ALPS circuits.
`Router(config)# alps enable-alarms peer [ip-address]`	Enables alarms for the ALPS peers.

Updating a Circuit

You can clear or update the circuits on the ALPS network. If a specific name is entered, the update action will be executed only on a configured circuit with that name; otherwise, the action will be performed on all configured circuits. If the circuit uses the ATP protocol, an update consists of a closing and reopening

of the ALPS circuit (the same action performed when clearing the circuit). If the circuit is a MATIP circuit, the update results in the sending of a configuration update (in the form of a MATIP Session Open command). You can update the circuit only on enabled or active (opened or opening state) ALPS circuits.

To update one or more ALPS circuits, use the following command in EXEC mode:

Command	Purpose
Router# **alps update-circuit** [*name*]	Specifies name of circuit to update.

Verifying ALPS

Perform the tasks in the following steps to verify the components of the ALPS network:

Step 1 Verify that the connection between the router and the ASCU is up by polling the ASCU. Enter the **show alps ascu** command and check the state field. UP indicates that the ASCU is responding to the polling. DOWN indicates that the connection is not responding to the polling.

```
router# show alps ascu

interface    dlc   id   a1   a2   circuit    pkt_tx  pkt_rx  state
-------------------------------------------------------------------------------
Serial6      ALC   42   60   70   CKT_ALC_1  416     416     UP
Serial6      ALC   45   60   72   CKT_ALC_1  600     600     UP
Serial6      ALC   48   62   78   CKT_ALC_2  0       0       DOWN
Serial7      UTS   21   22   13   CKT_UTS    4830    4830    UP
```

Step 2 Verify that the peer between the router and the host is connected. Enter the **show alps peer** command and check the state field. OPENED indicates that the circuit is connected. DISCONN indicates that the circuit is disconnected.

```
router# show alps peers

local_peer : ip_address = 192.168.25.2

ip_address     conn_id             state    pkt_t  pkt_rx
-------------------------------------------------------------------------------
192.168.20.3   MATIP_A_CKT_UTS     OPENED   1023   1023
192.168.70.2   MATIP_A_CKT_ALC_1   OPENED   4852   4757
192.168.70.2   MATIP_A_CKT_ALC_2   OPENED   1      1
192.168.70.3   MATIP_A_CKT_ALC_1   DISCONN  0      0
192.168.70.3   MATIP_A_CKT_ALC_2   DISCONN  0      0
```

Step 3 Verify that the ALPS circuit to the peer host is open and connected. Enter the **show alps circuits** command and check the state field. OPEN indicates that the circuit is connected. INOP indicates that the circuit is disconnected.

```
router# show alps circuits

name         pri_peer       curr_peer      dlc   state  pkt_tx  pkt_rx
-------------------------------------------------------------------------------
ALC_EMTOX    192.168.45.2   192.168.45.2   ALC   OPEN   944     944
UTS_AX25     192.168.45.2   192.168.45.2   UTS   OPEN   425     425
```

Monitoring and Maintaining ALPS

To monitor the status of the ALPS feature, use the following commands in EXEC mode:

Command	Purpose
Router# **show alps ascu** [*interface*] [*id*] [**detail**]	Displays the status of the ALPS ASCU.
Router# **show alps circuits** [**peer** *ip address*] [**name** *name*] [**detail**]	Displays the status of the ALPS circuits.
Router# **show alps peers** [**ipaddress** *addr*] [**detail**]	Displays the status of the ALPS remote peers.

ALPS Configuration Examples

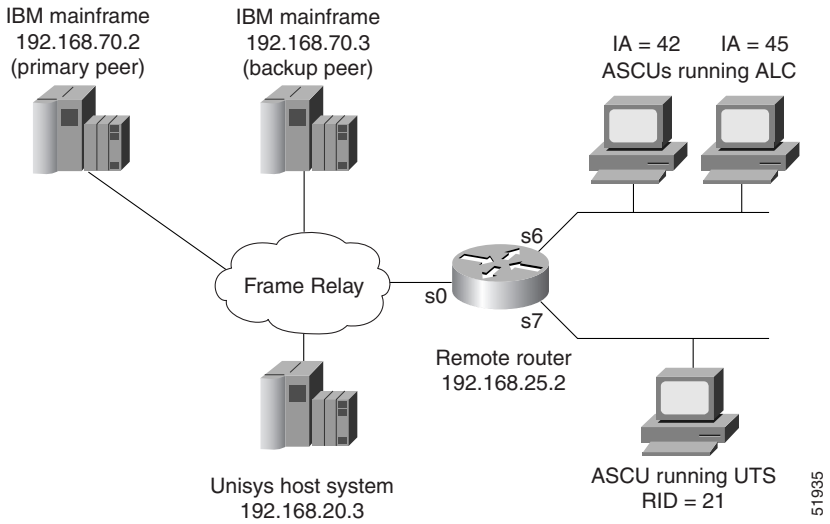
This section provides the following configuration examples:

- ALPS with MATIP Configuration for ALC and UTS Example, page 15
- ALPS Configuration for ALC and AX.25 Example, page 17
- ALPS Configuration for UTS and EMTOX Example, page 19

ALPS with MATIP Configuration for ALC and UTS Example

Figure 216 shows a simple example of a router topology for the ALPS with MATIP feature. The configuration corresponding to this topology follows.

Figure 216 Router Topology for the ALPS with MATIP Configuration Example



IA = interchange address
RID = remote identifier

ALC/UTS Router Configuration

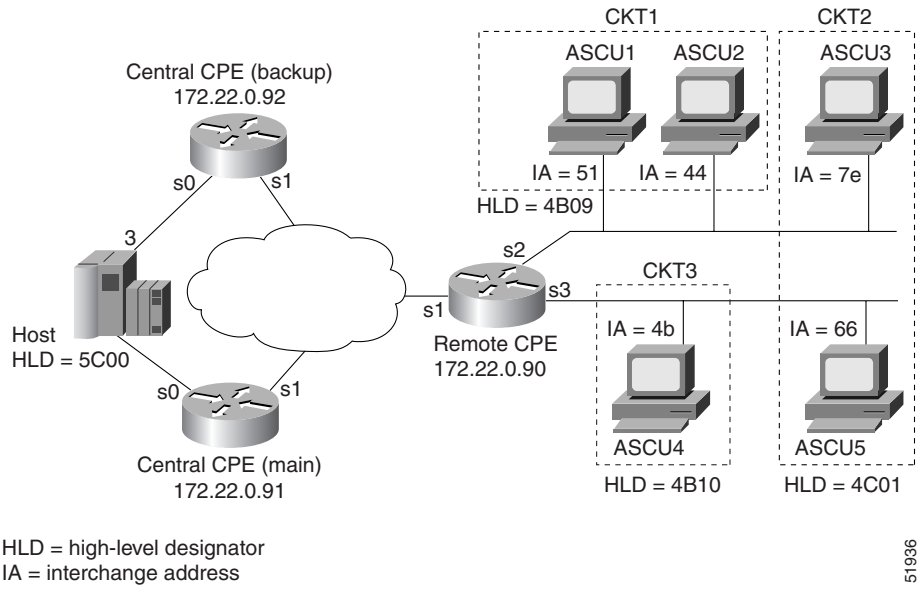
```
(config)# hostname alps-rcpe
(config)# alps local-peer 192.168.25.2
(config)# alps keepalive interval 45 retry 2
(config)# alps remote-peer 192.168.20.3 protocol matip-a dynamic status-interval 60
(config)# alps remote-peer 192.168.70.2 protocol matip-a dynamic 0 no-circuit 10
(config)# alps remote-peer 192.168.70.3 protocol matip-a dynamic 45
(config)# alps enable-alarms peer 192.168.70.2
(config)# alps enable-alarms ascu
!
(config)# alps circuit CKT_ALC_1
(config-alps-circ)# alps primary-peer 192.168.70.2 backup-peer 192.168.70.3
(config-alps-circ)# alps connection-type permanent
(config-alps-circ)# alps local-hld 2525
(config-alps-circ)# alps enable-circuit
!
(config)# alps circuit CKT_UTS
(config-alps-circ)# alps primary-peer 192.168.20.3
(config-alps-circ)# alps mpx single
(config-alps-circ)# alps idle-timer 90
(config-alps-circ)# alps local-hld 2527
(config-alps-circ)# alps enable-circuit
(config-alps-circ)# alps service-msg-interval 2
!
(config)# interface Loopback0
(config-if)# ip address 192.168.25.2 255.255.255.0


(config)# interface Serial0
(config-if)# ip address 210.100.50.2 255.255.255.0
```

```
(config-if)# encapsulation frame-relay IETF
(config-if)# frame-relay map ip 210.100.60.2 40
(config-if)# frame-relay map ip 210.100.70.2 50
!
(config)# interface Serial6
(config-if)# encapsulation alc
(config-if)# alps t1 6
(config-if)# alps t2 8
(config-if)# alps poll-pause 100
(config-if)# clockrate 9600
!
(config-if)# alps ascu 42
(config-alps-ascu)# alps default-circuit CKT_ALC_1
(config-alps-ascu)# alps a1-map 60 a2-map 70
(config-alps-ascu)# alps enable-ascu
!
(config-if)# alps ascu 45
(config-alps-ascu)# alps default-circuit CKT_ALC_1
(config-alps-ascu)# alps a1-map 60 a2-map 72
(config-alps-ascu)# alps enable-ascu
!
(config)# interface Serial7
(config-if)# encapsulation uts
(config-if)# alps n3 4
(config-if)# alps poll-pause 125
(config-if)# clockrate 4800
!
(config-if)# alps ascu 21
(config-alps-ascu)# alps default-circuit CKT_UTS
(config-alps-ascu)# alps a1-map 22 a2-map 13
(config-alps-ascu)# alps enable-ascu
!
```

ALPS Configuration for ALC and AX.25 Example

Figure 217 shows a simple router topology for the ALPS feature with ALC encapsulation. The configuration for this topology follows.

Figure 217 Router Topology for the ALPS Configuration for ALC Encapsulation Example



Remote CPE Configuration

```
(config)# alps local-peer 172.22.0.90
(config)# alps keepalive interval 60
(config)# alps remote-peer 172.22.0.91
(config)# alps remote-peer 172.22.0.92 dynamic 60
(config)# alps service-msg-list 1 number 2 TERMINAL OFF
!
(config)# alps circuit CKT1
(config-alps-circ)# alps primary-peer 172.22.0.91 backup-peer 172.22.0.92
(config-alps-circ)# alps local-hld 4B09 remote-hld 5C00
(config-alps-circ)# alps connection-type permanent 30
(config-alps-circ)# alps lifetime-timer 3
(config-alps-circ)# alps hostlink 3 ax25 120 winout 3 winin 3
(config-alps-circ)# alps service-msg-interval 3
(config-alps-circ)# alps service-msg-list 1
(config-alps-circ)# alps enable-circuit
!
(config)# alps circuit CKT2
(config-alps-circ)# alps primary-peer 172.22.0.91 backup-peer 172.22.0.92
(config-alps-circ)# alps local-hld 4C01 remote-hld 5C00
(config-alps-circ)# alps hostlink 3 ax25 1500 winout 4 winin 5
(config-alps-circ)# alps enable-circuit
!


(config)# alps circuit CKT3
(config-alps-circ)# alps primary-peer 172.22.0.91
(config-alps-circ)# alps local-hld 4B10 remote-hld 5C00
(config-alps-circ)# alps connection-type permanent 30
(config-alps-circ)# alps lifetime-timer 6
```

```
(config-alps-circ)# alps hostlink 3 ax25 905
(config-alps-circ)# alps enable-circuit
!
(config)# interface serial 1
(config-if)# ip address 172.22.0.90 255.255.255.0
!
(config)# interface serial 2
(config-if)# encapsulation alc
(config-if)# alps t1 3
(config-if)# alps t2 6
(config-if)# alps n1 3
(config-if)# alps n2 2
(config-if)# alps servlim 20
!
(config-if)# alps ascu 51
(config-alps-ascu)# alps default-circuit CKT1
(config-alps-ascu)# alps a1-map 40 a2-map 2D
(config-alps-ascu)# alps retry-option resend
(config-alps-ascu)# alps max-msg-length 1950
(config-alps-ascu)# alps error-display 6d 78
(config-alps-ascu)# alps enable-ascu
!
(config-if)# alps ascu 44
(config-alps-ascu)# alps default-circuit CKT1
(config-alps-ascu)# alps a1-map 40 a2-map 2E
(config-alps-ascu)# alps max-msg-length 590
(config-alps-ascu)# alps error-display 6d 78
(config-alps-ascu)# alps enable-ascu
!
(config-if)# alps ascu 7E
(config-alps-ascu)# alps default-circuit CKT2
(config-alps-ascu)# alps a1-map 40 a2-map 2F
(config-alps-ascu)# alps retry-option re-send
(config-alps-ascu)# alps max-msg-length 2000
(config-alps-ascu)# alps error-display 6d 78
(config-alps-ascu)# alps enable-ascu

(config)# interface serial 3
(config-if)# encapsulation alc
(config-if)# alps t1 5
(config-if)# alps t2 6
(config-if)# alps n1 1
(config-if)# alps n2 2
(config-if)# alps servlim 20
!
(config-if)# alps ascu 4B
(config-alps-ascu)# alps default-circuit CKT3
(config-alps-ascu)# alps a1-map 63 a2-map 41
(config-alps-ascu)# alps retry-option re-send
(config-alps-ascu)# alps max-msg-length 1960
(config-alps-ascu)# alps error-display 6d 78
(config-alps-ascu)# alps enable-ascu

(config-if)# alps ascu 66
(config-alps-ascu)# alps default-circuit CKT2
(config-alps-ascu)# alps a1-map 71 a2-map 21
(config-alps-ascu)# alps max-msg-length 3800
(config-alps-ascu)# alps error-display 6d 78
(config-alps-ascu)# alps enable-ascu
```

Central CPE Configuration (Main)

AX.25 Host

```
(config)# alps local-peer 172.22.0.91 promiscuous
(config)# interface serial 0
(config-if)# encapsulation x25 ax25
(config-if)# x25 ltc 1024
(config-if)# alps host-hld 5C00 host-link 3 ax25
```

Central CPE Configuration (Backup)

AX.25 Host

```
(config)# alps local-peer 172.22.0.92 promiscuous
(config)# interface serial 0
(config-if)# encapsulation x25 ax25
(config-if)# x25 ltc 1024
(config-if)# alps host-hld 5C00 host-link 3 ax25
```

ALPS Configuration for UTS and EMTOX Example

The following configuration is an example of routing P1024C UTS data frames across the network between central and remote equipment.

Remote Router Configuration

```
(config)# hostname alps-rcpe
(config)# alps local-peer 200.100.25.2
(config)# alps keepalive interval 45 retry 5
(config)# alps remote-peer 200.100.40.2
(config)# alps enable-alarms peer 200.100.40.2
(config)# alps enable-alarms ascu

(config)# alps circuit UTS_EMTOX
(config-alps-circ)# alps primary-peer 200.100.40.2
(config-alps-circ)# alps idle-timer 90
(config-alps-circ)# alps local-hld 2525 remote-hld 5050
(config-alps-circ)# alps mpx single
(config-alps-circ)# alps hostlink 6 emtox 1100 ops 512 ips 512
(config-alps-circ)# alps service-msg-interval 2
(config-alps-circ)# alps enable-circuit

(config)# interface Loopback0
(config-if)# ip address 200.100.25.2 255.255.255.0

(config)# interface Serial0
(config-if)# ip address 200.100.50.2 255.255.255.0
(config-if)# encapsulation frame-relay IETF
(config-if)# frame-relay map ip 200.100.50.3 20


(config)# interface Serial1
(config-if)# encapsulation uts
(config-if)# alps n1 5
(config-if)# alps n3 4
(config-if)# alps poll-pause 200
(config-if)# clockrate 4800
!
(config-if)# alps ascu 21
(config-alps-ascu)# alps default-circuit UTS_EMTOX
(config-alps-ascu)# alps a1-map 22 a2-map 13
```

```
(config-alps-ascu)# alps enable-ascu
!
```

Central CPE Configuration

```
(config)# hostname alps-ccpe
(config)# alps local-peer 200.100.40.2 promiscuous
(config)# alps enable-alarms circuit
!
(config)# interface Loopback0
(config-if)# ip address 200.100.40.2 255.255.255.0
!
(config)# interface Serial0
(config-if)# ip address 200.100.50.3 255.255.255.0
(config-if)# encapsulation frame-relay IETF
(config-if)# clockrate 56000
(config-if)# frame-relay map ip 200.100.50.2 20
!
(config)# interface Serial2
(config-if)# encapsulation x25 dce
(config-if)# alps host-hld 5050 host-link 6 emtox 2222
(config-if)# alps translate 110* 200.100.25.2
(config-if)# clockrate 64000
```

Configuring DSPU and SNA Service Point Support

This chapter describes Cisco IOS support for Systems Network Architecture (SNA) downstream physical unit (DSPU) devices and SNA Service Point. For a complete description of the DSPU and SNA Service Point commands mentioned in this chapter, refer to the "DSPU and SNA Service Point Configuration Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

DSPU is a software feature that enables the router to function as a physical unit (PU) concentrator for SNA PU type 2 nodes. PU concentration at the device simplifies the task of PU definition at the upstream host while providing additional flexibility and mobility for downstream PU devices.

The DSPU feature allows you to define downstream PU type 2 devices in the Cisco IOS software. DSPU reduces the complexity of host configuration by letting you replace multiple PU definitions that represent each downstream device with one PU definition that represents the router.

Because you define the downstream PUs at the router rather than the host, you isolate the host from changes in the downstream network topology. Therefore you can insert and remove downstream PUs from the network without making any changes on the host.

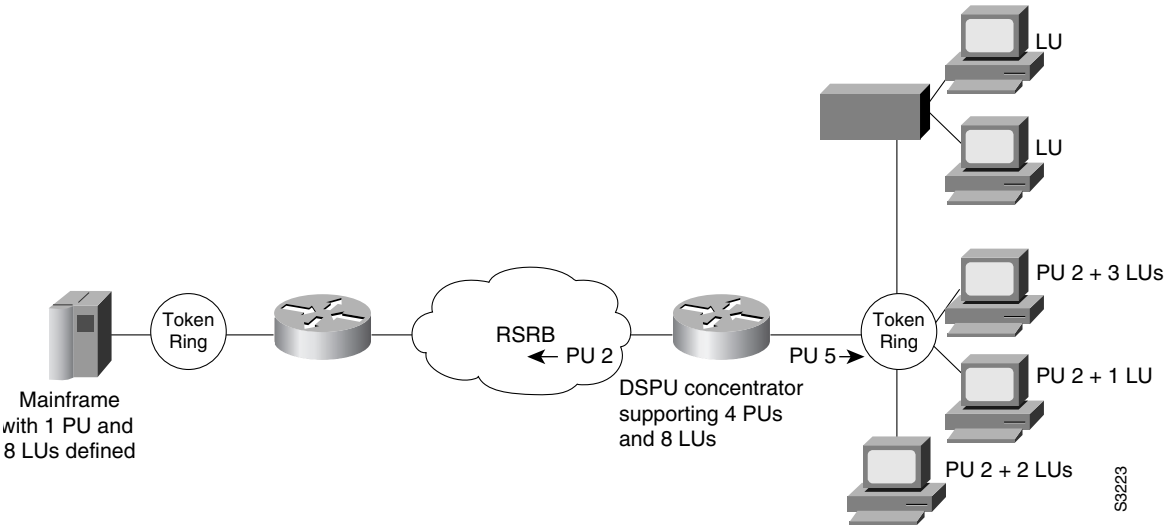
Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

The concentration of downstream PUs at the router also reduces network traffic on the WAN by limiting the number of sessions that must be established and maintained with the host. The termination of downstream sessions at the router ensures that idle session traffic does not appear on the WAN.

SNA service point support in the Cisco IOS software assumes that NetView or an equivalent product is available at the SNA host. The user interacts with the network management feature in the router and at the SNA host. In the Cisco IOS software, you can configure the host connection and show the status of this connection. At the SNA host, you can use the NetView operator's console to view alerts and to send and receive Cisco syntax commands to the Cisco device.

Figure 218 shows a router functioning as a DSPU concentrator.

Figure 218 Router Acting as a DSPU Concentrator

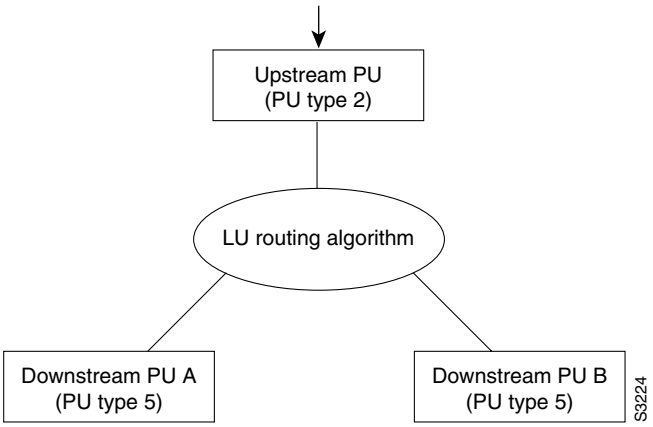


Typically, a router establishes one or more upstream connections with one or more hosts and many downstream connections with PU type 2 devices. From an SNA perspective, the router appears as a PU type 2 device to the upstream host and assumes the role of a system services control point (SSCP) appearing as a PU type 5 device to its downstream PUs.

The SSCP sessions established between the router and its upstream host are completely independent of the SSCP sessions established between the router and its downstream PUs. SNA traffic is routed at a logical unit (LU) level using a routing algorithm that maps downstream LUs onto upstream LUs.

Figure 219 illustrates the SNA perspective of DSPU.

Figure 219 SNA Perspective of DSPU



DSPU Configuration Task List

To configure DSPU, perform the tasks in the following sections:

- Defining DSPU Upstream Hosts, page 3 (Required)
- Defining Downstream PUs, page 4 (Required)
- Defining DSPU LUs, page 7 (Required)
- Configuring DSPU to Use a Data-Link Control, page 8 (Optional)
- Defining the Number of Outstanding, Unacknowledged Activation RUs, page 16 (Optional)

See the "DSPU and SNA Service Point Configuration Examples" section on page 24 for examples.

Defining DSPU Upstream Hosts

The upstream host provides logical units (LUs) that the Cisco IOS software assigns for use by its downstream PUs. Because one upstream host can only provide a maximum of 255 LUs, the DSPU feature supports multiple hosts. Multiple upstream host support allows the DSPU router to provide more than 255 LUs for use by its downstream PUs.

To define a DSPU host over Token Ring, Ethernet, Fiber Distributed Data Interface (FDDI), remote source-route bridging (RSRB), or virtual data-link control connections, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu host` *host-name* **xid-snd** *xid* **rmac** *remote-mac* [**rsap** *remote-sap*] [**lsap** *local-sap*] [**interface** *slot/port*] [**window** *window-size*] [**maxiframe** *max-iframe*] [**retries** *retry-count*] [**retry-timeout** *retry-timeout*] [**focalpoint**]	Defines a DSPU host over Token Ring, Ethernet, FDDI, RSRB, or virtual data-link control connections.

To define a DSPU host over a Synchronous Data-Link Control (SDLC) connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu host host-name xid-snd xid sdlc sdlc-addr [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a DSPU host over an SDLC connection.

To define a DSPU host over an X.25/Qualified Logical Link Control (QLLC) connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu host host-name xid-snd xid x25 remote-x121-addr [qllc local-x121-subaddr] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a DSPU host over an X.25/QLLC connection.

To define a DSPU host over a Frame Relay connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu host host-name xid-snd xid dlci dlci-number [rsap remote-sap] [lsap local-sap] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a DSPU host over a Frame Relay connection.

Defining Downstream PUs

To define the downstream PUs, perform either of the tasks in the following sections, depending on your circumstances:

- Explicitly Defining a Downstream PU, page 4
- Enabling the Default PU Option, page 7

Explicitly Defining a Downstream PU

Explicitly define a downstream PU if you require the Cisco IOS software to perform verification checking on incoming downstream connections or to initiate an outgoing downstream connection.

For Cisco IOS Release 11.3 and later releases, the number of DSPU PUs you can configure is 1024.

To explicitly define a downstream PU over Token Ring, Ethernet, FDDI, RSRB, virtual data-link control, or native client interface architecture (NCIA) connections, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pu pu-name [rmac remote-mac] [rsap remote-sap] [lsap local-sap] [xid-rcv xid] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout]`	Explicitly defines a downstream PU over Token Ring, Ethernet, FDDI, RSRB, virtual data-link control, or NCIA connections.

To explicitly define a downstream PU over an SDLC connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pu pu-name [sdlc sdlc-addr] [xid-rcv xid] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout]`	Explicitly defines a downstream PU over an SDLC connection.

To explicitly define a downstream PU over an X.25/QLLC connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pu` *pu-name* [**x25** *remote-x121-addr*] [**qllc** *local-x121-subaddr*] [**xid-rcv** *xid*] [**interface** *slot/port*] [**window** *window-size*] [**maxiframe** *max-iframe*] [**retries** *retry-count*] [**retry-timeout** *retry-timeout*]	Explicitly defines a downstream PU over an X.25/QLLC connection.

To explicitly define a downstream PU over a Frame Relay connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pu` *pu-name* [**dlci** *dlci-number*] [**rsap** *remote-sap*] [**lsap** *local-sap*] [**xid-rcv** *xid*] [**interface** *type slot/port*] [**window** *window-size*] [**maxiframe** *max-iframe*] [**retries** *retry-count*] [**retry-timeout** *retry-timeout*]	Explicitly defines a downstream PU over a Frame Relay connection.

A PU definition must have either an xid-rcv parameter or an address (rmac, sdlc, x25 or dlci) parameter.

If the Cisco IOS software will perform verification checking on incoming downstream connections, there are several combinations of parameters that you can configure for verification matching. Note that the address parameter, when specified, is considered to be the primary key on the PU definition. Therefore, if both an address and xid-rcv are configured, the matching algorithm will match on the address and ignore the xid-rcv parameter.

- Match on xid-rcv value only

  User may define a downstream PU using only the xid-rcv value so that any connecting PU that specifies the value of the configured XID will match that PU definition.

- Match on xid-rcv and interface values

  User may define a downstream PU using the xid-rcv and interface values so that any PU connecting into the configured interface that specifies the value of the configured XID will match the PU definition.

- Match on addressing values only

  User may define a downstream PU using only the addressing values (RMAC/RSAP/LSAP, SDLC, DLCI/RSAP/LSAP, or X25/QLLC) so that any connecting PU with addressing that matches the configured addressing will match that PU definition. If no PU definition is found to match the incoming RSAP, then a match is accepted on a PU that has the correct RMAC/LSAP or DLCI/LSAP.

- Match on addressing and interface values

  User may define a downstream PU using the interface and addressing values (RMAC/RSAP/LSAP, SDLC, DLCI/RSAP/LSAP, or X25/QLLC) so that any PU connecting into the configured interface with addressing that matches the configured addressing will match the PU definition. If no PU definition is found to match the incoming RSAP, then a match is accepted on a PU that has the correct RMAC/LSAP or DLCI/LSAP and interface.

The Cisco IOS software rejects any incoming downstream connections that do not match the parameters of a defined downstream PU unless the default PU option is also enabled.

Enabling the Default PU Option

Configure the DSPU default PU option if you do not require the Cisco IOS software to verify incoming downstream connections. The default PU option allows the software to accept incoming downstream connections without an explicit definition for the downstream PU.

To enable the default PU option, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu default-pu [window window-size] [maxiframe max-iframe]`	Enables the default PU option.

Defining DSPU LUs

Specify the LU routing algorithm used to map the upstream LUs to the downstream LUs and to define all LUs for each upstream and downstream PU.

The DSPU feature assigns upstream LUs to downstream LUs based on the selected LU routing algorithm and performs the mapping necessary for SNA data transfer.

The DSPU feature supports two alternative mapping algorithms that are described in the following sections:

- Defining Dedicated LU Routing, page 7
- Defining Pooled LU Routing, page 7

An upstream host PU or downstream PU can support up to 255 LU sessions. The DSPU feature allows each LU to be individually configured for either dedicated LU routing or pooled LU routing.

Defining Dedicated LU Routing

You can configure an upstream LU so that it is reserved, or dedicated, for use by a specific downstream LU.

To define a dedicated LU or a range of dedicated LUs for an upstream host and downstream PU, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu lu lu-start [lu-end] {host host-name host-lu-start | pool pool-name} [pu pu-name]`	Defines a dedicated LU or a range of dedicated LUs for a downstream PU.

See the "Dedicated LU Routing Example" section on page 24 for an example of dedicated LU routing.

Defining Pooled LU Routing

You can configure an upstream host LU so that it is a member of a pool of LUs. When a downstream connection is established and the downstream LU is configured as a pooled LU, the Cisco IOS software selects an upstream LU from the pool for assignment to the downstream LU.

Pooled LU routing allows a limited number of upstream host LUs to be shared (at different times) among many downstream LUs.

To define a host LU or a range of host LUs in an LU pool, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu pool pool-name host host-name lu lu-start [lu-end] [inactivity-timeout inactivity-minutes]`	Defines a host LU or a range of host LUs in an LU pool.

You can configure a downstream LU as a pooled LU. When a downstream connection is established and the downstream LU is configured as a pooled LU, the software selects an upstream LU from the specified pool for assignment to the downstream LU.

To define a pooled LU or a range of pooled LUs for a downstream PU, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu lu lu-start [lu-end] pool pool-name pu pu-name`	Defines a pooled LU or a range of pooled LUs for a downstream PU.

See the "Pooled LU Routing Example" section on page 25 for an example of pooled LU routing.

Configuring DSPU to Use a Data-Link Control

The final step in configuring DSPU is to define the data-link controls that will be used for upstream host and downstream PU connections.

The DSPU feature supports the data-link controls described in the following sections:

Configuring DSPU to Use Token Ring, Ethernet, or FDDI

You can configure DSPU to use the Token Ring, Ethernet, or FDDI data-link controls by enabling a service access point (SAP) address on the interface. Each interface can support up to 254 local SAPs enabled for either upstream or downstream connections; a local SAP cannot be enabled for both upstream and downstream connections on the same interface.

To enable a local SAP on the Token Ring, Ethernet, or FDDI interfaces for use by upstream hosts, use the following command in interface configuration mode:

Command	Purpose
Router(config-if)# **dspu enable-host** [**lsap** *local-sap*]	Enables local SAP for upstream hosts.

To enable a local SAP on the Token Ring, Ethernet, or FDDI interfaces for use by downstream PUs, use the following command in interface configuration mode:

Command	Purpose
Router(config-if)# **dspu enable-pu** [**lsap** *local-sap*]	Enables local SAP for downstream PUs.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote device (upstream host or downstream PU). Alternately, initiate an outgoing connection to the remote device by using the following command in interface configuration mode:

Command	Purpose
Router(config-if)# **dspu start** {*host-name* | *pu-name*}	Initiates a connection with an upstream host or a downstream PU via Token Ring or Ethernet.

Configuring DSPU to Use RSRB

To configure DSPU to use RSRB, you must create a DSPU/RSRB data-link control.

Cisco's implementation of DSPU/RSRB data-link control uses the concept of a virtual Token Ring device residing on a virtual Token Ring to represent the Cisco IOS software to upstream hosts and downstream PUs across an RSRB network. This is similar to Cisco's implementation of SDLLC.

Because the upstream host and downstream PU expects its peer to also be on a Token Ring, you must assign a virtual Token Ring address (the DSPU virtual MAC address) to the DSPU/RSRB data-link control. Like real Token Ring addresses, the DSPU virtual MAC address must be unique across the network.

In addition to assigning the DSPU virtual MAC address, you must also assign a DSPU virtual ring number to the DSPU/RSRB data-link control. The DSPU virtual ring number must be unique across the network.

Note The DSPU virtual ring number is a different number from the virtual ring group numbers that you use to configure RSRB and multiport bridging.

The combination of the DSPU virtual MAC address and the DSPU virtual ring number identifies the DSPU/RSRB data-link control interface to the rest of an RSRB network.

When an end station (either an upstream host or a downstream PU) attempts to connect with the DSPU software, the following events occur:

1. The end station sends explorer packets with the locally administered MAC address on the router interface to which the end station is connected.
2. The router configured with that locally administered MAC address or with the hardware MAC address intercepts the frame, fills in the DSPU virtual ring number and the DSPU bridge number in the routing information field (RIF), and sends a response to the end station.
3. The end station establishes a session with the DSPU router.

To define the DSPU/RSRB data-link control interface, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines an RSRB ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group tcp ip-address local-ack`	Defines a remote peer with the local acknowledgment feature.
Step 3	`Router(config)# dspu rsrb local-virtual-ring bridge-number target-virtual-ring virtual-macaddr`	Defines the DSPU/RSRB interface.

After you define the DSPU RSRB data-link control, configure DSPU to use the RSRB data-link control by enabling a local SAP for either upstream or downstream connections.

To enable a local SAP on RSRB for use by upstream hosts, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu rsrb enable-host [lsap local-sap]`	Enables local SAP for upstream hosts.

To enable a local SAP on RSRB for use by downstream PUs, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu rsrb enable-pu [lsap local-sap]`	Enables local SAP for downstream PUs.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote device (upstream host or downstream PU) over RSRB. Alternatively, initiate an outgoing connection to the remote device by using the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu rsrb start {host-name | pu-name}`	Initiates a connection with an upstream host or a downstream PU via RSRB.

Configuring DSPU to Use RSRB with Local Acknowledgment

Configuring DSPU to use RSRB with local acknowledgment is identical to configuring RSRB with local acknowledgment. If you add the **local-ack** keyword to the **source-bridge remote-peer** configuration command, DSPU will use local acknowledgment for any end stations that connect to DSPU from that peer.

To configure DSPU to use RSRB with local acknowledgment, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines an RSRB ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group tcp ip-address local-ack`	Defines a remote peer with the local acknowledgment feature.
Step 3	`Router(config)# dspu rsrb local-virtual-ring bridge-number target-virtual-ring virtual-macaddr`	Defines the DSPU/RSRB interface.

Configuring DSPU to Use Virtual Data-Link Control

To configure DSPU to use virtual data-link control, you must create a DSPU virtual data-link control interface.

Similar to our implementation of SDLLC, the DSPU virtual data-link control interface uses the concept of a virtual Token Ring device residing on a virtual Token Ring to represent the Cisco IOS software to upstream hosts and downstream PUs across a network.

Because the upstream host and downstream PU expects its peer to also be on a Token Ring, you must assign a virtual Token Ring address (the DSPU virtual MAC address) to the DSPU virtual data-link control interface. Like real Token Ring addresses, the DSPU virtual MAC address must be unique across the network.

In addition to assigning the DSPU virtual MAC address, you must also identify the source-route bridging virtual ring number with which the DSPU virtual MAC address will be associated. The source-route bridging virtual ring number is set using the **source-bridge ring-group** command. This is documented in the "Source-Route Bridging Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

The combination of the DSPU virtual MAC address and the source-route bridging virtual ring number identifies the DSPU virtual data-link control interface to the rest of the DLSw+ network.

When an end station (either an upstream host or a downstream PU) attempts to connect with the DSPU software, the following events occur:

1. The end station sends explorer packets with the locally administered MAC address on the router interface to which the end station is connected.
2. The router configured with that locally administered MAC address intercepts the frame, DLSw+ adjusts the routing information field (RIF), and sends a response to the end station.
3. The end station establishes a session with the DSPU router.

Prior to creating the DSPU virtual data-link control interface, you must also configure DLSw+ peers so that DLSw+ can provide the communication path. The commands for defining DLSw+ local and remote peers are documented in the "DLSw+ Configuration Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

To define the DSPU virtual data-link control interface, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu vdlc` *ring-group virtual-mac-address*	Defines the DSPU virtual data-link control interface.

After you define the DSPU virtual data-link control interface, configure DSPU to use virtual data-link control by enabling a local SAP for either upstream or downstream connections.

To enable a local SAP on the virtual data-link control for use by upstream hosts, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu vdlc enable-host [lsap` *local-sap*`]`	Enables local SAP for upstream hosts.

To enable a local SAP on the virtual data-link control for use by downstream PUs, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu vdlc enable-pu [lsap` *local-sap*`]`	Enables local SAP for downstream PUs.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote device (upstream host or downstream PU) using virtual data-link control. Alternately, initiate an outgoing connection to the remote device by using the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu vdlc start {`*host-name* `|` *pu-name*`}`	Initiates a connection with an upstream host or a downstream PU via virtual data-link control.

Configuring DSPU to Use SDLC

Before DSPU may be configured to use the SDLC data-link control, the serial interface must be defined for SDLC encapsulation and assigned an SDLC role.

To define the serial interface to use SDLC and specify the SDLC role, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# encapsulation sdlc`	Enables SDLC encapsulation on the serial interface.
Step 2	`Router(config-if)# sdlc role {none | primary | secondary | prim-xid-poll}`	Specifies the SDLC role of the router.

For the connection to be established without XID exchange, the SDLC role must be **primary** if DSPU will be initiating connections to the SDLC partner. The SDLC role must be **secondary** or **none** if the SDLC partner will be initiating connections with DSPU.

When an XID exchange is required, the SDLC role must be **prim-xid-poll** or **none** if DSPU will be initiating connections to the SDLC partner. The role must be **none** if the SDLC partner will be initiating connections with DSPU.

The SDLC addresses used on the SDLC link must also be defined. If DSPU is configured to initiate the connection, then the SDLC address identifies the SDLC partner. If the remote SDLC device initiates the connection, then the SDLC address identifies the address for which a connection will be accepted.

To configure the SDLC address, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sdlc address hexbyte`	Defines the SDLC address.

Finally, the SDLC address must be enabled for use by DSPU. Each interface can support up to 255 SDLC addresses enabled for either upstream or downstream connections; an SDLC address cannot be enabled for both upstream and downstream connections on the same interface. If the SDLC role is **none**, there can be only one SDLC address on that interface.

To enable an SDLC address for use by upstream host connections, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-host sdlc sdlc-address`	Enables the SDLC address for an upstream host.

To enable an SDLC address for use by downstream PU connections, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-pu sdlc sdlc-address`	Enables the SDLC address for the downstream PU.

When the SDLC role is configured as **primary**, DSPU initiates a connection with the remote device by sending set normal response mode (SNRM) when the SDLC address is enabled for DSPU.

When the SDLC role is configured as **prim-xid-poll**, DSPU initiates a connection with the remote device by sending a NULL XID when the SDLC address is enabled for DSPU.

When the SDLC role is configured as **secondary**, DSPU will not be ready to respond to SNRM until a **dspu start** *pu-name* command is issued.

When the SDLC role is configured as **none**, DSPU is ready to respond to a received XID or SNRM when the SDLC address is enabled for DSPU; otherwise, the connection may be initiated by issuing the **dspu start** *pu-name* command.

To configure DSPU to respond to SNRM when the SDLC role is configured as **secondary**, or to initiate a connection when the SDLC role is configured as **none**, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu start {host-name | pu-name}`	Initiates a connection with a remote device when the SDLC role is configured as **secondary** or **none**.

Configuring DSPU to Use QLLC

Before DSPU may be configured to use the QLLC data-link control, the serial interface must be defined for X.25 encapsulation and assigned an X.121 address.

To define the serial interface to use X.25, use the following commands in interface configuration mode:

Command	Purpose
`Router(config-if)# encapsulation x25 [dce]`	Enables X.25 encapsulation on the serial interface.
`Router(config-if)# x25 address x121-addr`	Defines an X.121 address.

X.25 routing must also be configured so that incoming calls to the local X.121 address can be appropriately routed to the serial interface and mapped into the QLLC data-link control.

To define X.25 routing, use the following commands in global configuration mode:

Command	Purpose
`Router(config)# x25 routing`	Enables X.25 routing.
`Router(config)# x25 route ^local-x121-addr.* alias serial slot/port`	Enables routing of X.25 packets to the serial interface.

To define which calls get mapped into QLLC, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# x25 map qllc x121-addr`	Defines the remote X.121 address for mapping into QLLC.

Finally, the local X.121 subaddress must be enabled for use by DSPU. An X.121 subaddress can be enabled for either upstream or downstream connections; an X.121 subaddress cannot be enabled for both upstream and downstream connections on the same interface.

To enable an X.121 subaddress for use by upstream host connections via QLLC, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-host qllc x121-subaddress`	Enables an X.121 subaddress for an upstream host.

To enable an X.121 subaddress for use by downstream PU connections via QLLC, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-pu qllc x121-subaddress`	Enables an X.121 subaddress for a downstream PU.

Once an X.121 subaddress is enabled, it is ready to accept incoming connection attempts from the remote device (upstream host or downstream PU) over QLLC. Alternatively, initiate an outgoing connection to the remote device by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu start {host-name | pu-name}`	Initiates a connection with an upstream host or a downstream PU via QLLC.

Configuring DSPU to Use Frame Relay

Before DSPU may be configured to use the LLC2/Frame Relay data-link control, the serial interface must be defined for Frame Relay encapsulation.

To define the serial interface for Frame Relay encapsulation, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# encapsulation frame-relay ietf`	Enables Frame Relay encapsulation on a serial interface.

The DLCI used on the Frame Relay link must be mapped into LLC2.

To configure the mapping of a DLCI into LLC2, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# frame-relay map llc2 dlci-number`	Configures DLCI mapping into LLC2.

Finally, the local SAP address must be enabled for use by DSPU. A SAP address can be enabled for either upstream or downstream connections; a SAP address cannot be enabled for both upstream and downstream connections on the same interface.

To enable a local SAP on the LLC2/Frame Relay interface for use by upstream hosts, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-host [lsap local-sap]`	Enables local SAP for upstream hosts.

To enable a local SAP for the LLC2/Frame Relay interface for use by downstream PUs, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu enable-pu [lsap local-sap]`	Enables local SAP for downstream PUs.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote device (upstream host or downstream PU) over Frame Relay. Alternatively, initiate an outgoing connection to the remote device by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# dspu start {host-name | pu-name}`	Initiates a connection with an upstream host or a downstream PU via LLC2 Frame Relay.

Configuring DSPU to Use NCIA

To configure DSPU to use NCIA, you must perform the following tasks:

- Configure the NCIA server as the underlying transport mechanism.
- Enable a local SAP on the NCIA server for use by downstream PUs.

To configure the NCIA server as the underlying transport mechanism, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu ncia server-number`	Configures the NCIA server as the underlying transport mechanism.

To enable a local SAP on the NCIA server for use by downstream PUs, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu ncia enable-pu [lsap local-sap]`	Enables local SAP for downstream PUs.

Defining the Number of Outstanding, Unacknowledged Activation RUs

The DSPU feature allows you to define the number of activation request/response units (RUs) such as ACTLUs or DDDLU NMVTs that can be sent by the Cisco IOS software before waiting for responses from the remote PU.

The DSPU activation window provides pacing to avoid depleting the router buffer pool during PU activation. Increasing the window size allows more LUs to become active in a shorter amount of time (assuming the required buffers for activation RUs are available). Decreasing the window size limits the amount of buffers the DSPU may use during PU activation. Typically, you do not need to change the default window size.

To define the number of unacknowledged activation RUs that can be outstanding, use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu activation-window window-size`	Defines the number of unacknowledged activation RUs.

Configuring SNA Service Point Support

Cisco's implementation of SNA Service Point support includes support for three commands: Alerts, RUNCMD, and Vital Product Data support.

Alert support is provided as the Cisco IOS software sends unsolicited Alerts to NetView (or an equivalent network management application) at the host. This function occurs at the various router interfaces and protocol layers within the device.

RUNCMD support enables you to send commands to the router from the NetView console using the NetView RUNCMD facility, and the router sends the relevant replies back to the RUNCMD screen. Some commands, such as **telnet**, **rsh**, **rlogin**, and **tn3270**, are not supported.

Vital Product Data support allows you to request Vital Product Data from the NetView console. The router replies to NetView with the relevant information.

To configure SNA Service Point support, perform the tasks in the following sections:

- Defining a Link to an SNA Host, page 17
- Configuring Service Point Support to Use a Data-Link Control, page 18
- Specifying Names for All Attached LANs, page 23
- Specifying the Physical Location of the Router, page 23

Note You must define the Service Point PU at the SNA host by using either ANS=STOP, or you can omit the ANS keyword. Do not use ANS=CONTINUE to define the Service Point PU at the SNA host. Coordinate this with your SNA host systems programmer.

Note You do not need to perform the tasks in the next section if you have configured a DSPU host with the **focalpoint** parameter.

Defining a Link to an SNA Host

To define a link to an SNA host over Token Ring, Ethernet, FDDI, RSRB, or virtual data-link control connections, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna host host-name xid-snd xid rmac remote-mac [rsap remote-sap] [lsap local-sap] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a link to an SNA host over Token Ring, Ethernet, FDDI, RSRB, or virtual data-link control connections.

To define a link to an SNA host over an SDLC connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna host host-name xid-snd xid sdlc sdlc-addr [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a link to an SNA host over an SDLC connection.

To define a link to an SNA host over an X.25/QLLC connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna host host-name xid-snd xid x25 remote-x121-addr [qllc local-x121-subaddr] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a link to an SNA host over an X.25/QLLC connection.

To define a link to an SNA host over a Frame Relay connection, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna host host-name xid-snd xid dlci dlci-number [rsap remote-sap] [lsap local-sap] [interface slot/port] [window window-size] [maxiframe max-iframe] [retries retry-count] [retry-timeout retry-timeout] [focalpoint]`	Defines a link to an SNA host over a Frame Relay connection.

Configuring Service Point Support to Use a Data-Link Control

To configure Service Point to use a data-link control, perform the tasks in one of the following sections:

- Configuring Service Point to Use Token Ring, Ethernet, or FDDI, page 19
- Configuring Service Point to Use RSRB, page 19
- Configuring Service Point to Use RSRB with Local Acknowledgment, page 19
- Configuring Service Point to Use Virtual Data-Link Control, page 20
- Configuring Service Point Support for Frame Relay, page 22
- Configuring Service Point Support for SDLC, page 22
- Configuring Service Point Support for X.25, page 22

Note You do not need to perform this task if you have configured a DSPU host with the **focalpoint** parameter and have configured the DSPU host to use a data-link control.

Configuring Service Point to Use Token Ring, Ethernet, or FDDI

To enable a local SAP on the Token Ring, Ethernet, or FDDI interfaces for use by SNA Service Point, use the following command in interface configuration mode:

:

Command	Purpose
`Router(config-if)# sna enable-host [lsap lsap-address]`	Enables local SAP for Service Point.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote host. Alternately, initiate an outgoing connection to the remote host by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# sna start host-name`	Initiates a connection with a host via Token Ring, Ethernet, or FDDI.

Configuring Service Point to Use RSRB

To define the Service Point/RSRB data-link control interface, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines an RSRB ring group.
Step 2	`Router(config)# sna rsrb local-virtual-ring bridge-number target-virtual-ring virtual-macaddr`	Defines the Service Point/RSRB interface.

To enable a local SAP on RSRB for use by hosts, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna rsrb enable-host [lsap local-sap]`	Enables local SAP for hosts.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote host over RSRB. Alternatively, initiate an outgoing connection to the remote host by using the following command in global configuration mode:

Command	Purpose
`Router(config)# sna rsrb start host-name`	Initiates a connection with a host via RSRB.

Configuring Service Point to Use RSRB with Local Acknowledgment

To configure Service Point to use RSRB with local acknowledgment, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# source-bridge ring-group ring-group [virtual-mac-address]`	Defines an RSRB ring group.
Step 2	`Router(config)# source-bridge remote-peer ring-group tcp ip-address local-ack`	Defines a remote peer with the local acknowledgment feature.
Step 3	`Router(config)# sna rsrb local-virtual-ring bridge-number target-virtual-ring virtual-macaddr`	Defines the Service Point/RSRB interface.

Configuring Service Point to Use Virtual Data-Link Control

To configure SNA Service Point to use virtual data-link control, you must create an SNA virtual data-link control interface.

Similar to our implementation of SDLLC, the SNA virtual data-link control interface uses the concept of a virtual Token Ring device residing on a virtual Token Ring to represent the Cisco IOS software to upstream hosts and downstream PUs across a network.

Because the upstream host and downstream PU expect their peer to also be on a Token Ring, you must assign a virtual Token Ring address (the SNA virtual data-link control virtual MAC address) to the SNA virtual data-link control interface. Like real Token Ring addresses, the SNA virtual MAC address must be unique across the network.

You must also identify the source-route bridging virtual ring number with which the SNA virtual MAC address will be associated. The source-route bridging virtual ring number is set using the **source-bridge ring-group** command, which is documented in the "Source-Route Bridging Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

The combination of the SNA virtual MAC address and the source-route bridging virtual ring number identifies the SNA virtual data-link control interface to the rest of the DLSw+ network.

When an end station (either an upstream host or a downstream PU) attempts to connect with the SNA Service Point software, the following events occur:

1. The end station sends explorer packets with the locally administered MAC address on the router interface to which the end station is connected.
2. The router configured with that locally administered MAC address intercepts the frame, DLSw+ adjusts the RIF and sends a response to the end station.
3. The end station establishes a session with the SNA Service Point router.

Prior to creating the SNA virtual data-link control interface, you must also configure DLSw+ peers so that DLSw+ can provide the communication path. The commands for defining DLSw+ local and remote peers are documented in the "DLSw+ Configuration Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 1 of 2).

To define the Service Point virtual data-link control interface, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna vdlc ring-group virtual-mac-address`	Defines the Service Point virtual data-link control interface.

After you create the SNA virtual data-link control interface, configure SNA Service Point to use virtual data-link control by enabling a local SAP for upstream connections. To enable a local SAP on virtual data-link control for use by hosts, use the following command in global configuration mode:

Command	Purpose
`Router(config)# sna vdlc enable-host [lsap local-sap]`	Enables local SAP for hosts.

Once a local SAP is enabled, it is ready to accept incoming connection attempts from the remote host using virtual data-link control. Alternatively, initiate an outgoing connection to the remote host by using the following command in global configuration mode:

Command	Purpose
`Router(config)# sna vdlc start host-name`	Initiates a connection with a host via virtual data-link control.

Configuring Service Point Support for Frame Relay

To configure Service Point support for Frame Relay, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# frame-relay map llc2 dlci-number`	Defines DLCI mapping into LLC2.
Step 2	`Router(config-if)# sna enable-host lsap lsap-address`	Enables a local SAP for hosts.

Configuring Service Point Support for SDLC

To configure Service Point support for SDLC, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# sdlc role {none | primary | secondary | prim-xid-poll}`	Specifies the SDLC role of the router.
Step 2	`Router(config-if)# sdlc address hexbyte`	Defines the SDLC address.
Step 3	`Router(config-if)# sna enable-host sdlc sdlc-address`	Enables the SDLC address for the host.

Configuring Service Point Support for X.25

To configure Service Point support for X.25, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# x25 address x121-address`	Defines an X.121 address.
Step 2	`Router(config-if)# x25 map qllc x121-addr`	Defines remote X.121 address for mapping to QLLC.
Step 3	`Router(config-if)# sna enable-host qllc x121-subaddress`	Enables QLLC subaddress for host.
Step 4	`Router(config-if)# x25 alias {destination-pattern | x121-address-pattern} [cud cud-pattern]`	Configures an interface X.25 alias address to accept calls with different destination addresses.

Specifying Names for All Attached LANs

You can specify names for all Token Ring or Ethernet LANs attached to the router. These names are used to identify the LAN when the Cisco IOS software sends an Alert to the host. To specify names for all attached LANs, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# lan-name lan-name`	Defines the name of an attached LAN.

Specifying the Physical Location of the Router

You can specify the physical location of the router if you intend requesting vital product information from the router. To specify the physical location, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# location location-description`	Defines the physical location of the router.

Monitoring and Maintaining DSPU and SNA Service Point Feature Status

You can monitor the status of the DSPU and SNA Service Point features. To display information about the state of the DSPU and SNA Service Point features, use the following commands in privileged EXEC mode:

Command	Purpose
`Router# show dspu`	Shows the status of all DSPU resources.
`Router# show dspu pu {host-name | pu-name} [all]`	Shows the status of DSPU hosts or downstream PUs.
`Router# show dspu pool pool-name [all]`	Shows the status of a DSPU pool.
`Router# show sna`	Shows the status of all SNA hosts.
`Router# show sna pu host-name [all]`	Shows the status of an SNA host.

To control the reporting of DSPU notification events (DSPU-specific SNMP Traps and Unsolicited SNA Messages to Operator), use the following command in global configuration mode:

Command	Purpose
`Router(config)# dspu notification-level {off | low | medium | high}`	Specifies the level of notification event reporting.

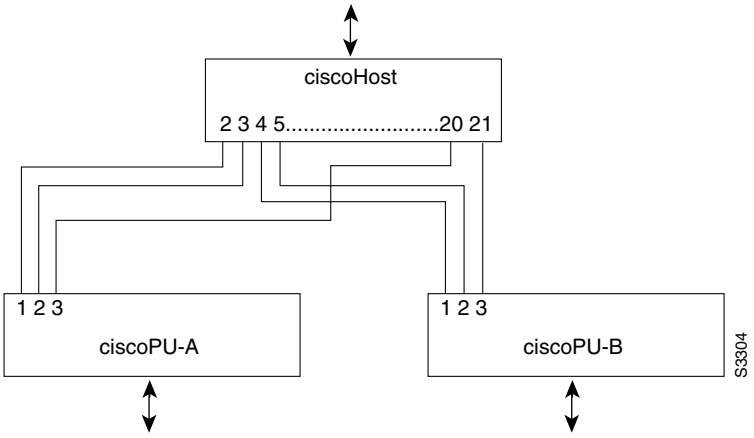
DSPU and SNA Service Point Configuration Examples

The following sections provide DSPU and SNA Service Point configuration examples:

Dedicated LU Routing Example

Figure 220 illustrates the use of dedicated LU routing. Each upstream host LU is dedicated for use by a specific downstream LU.

Figure 220 Dedicated LU Routing



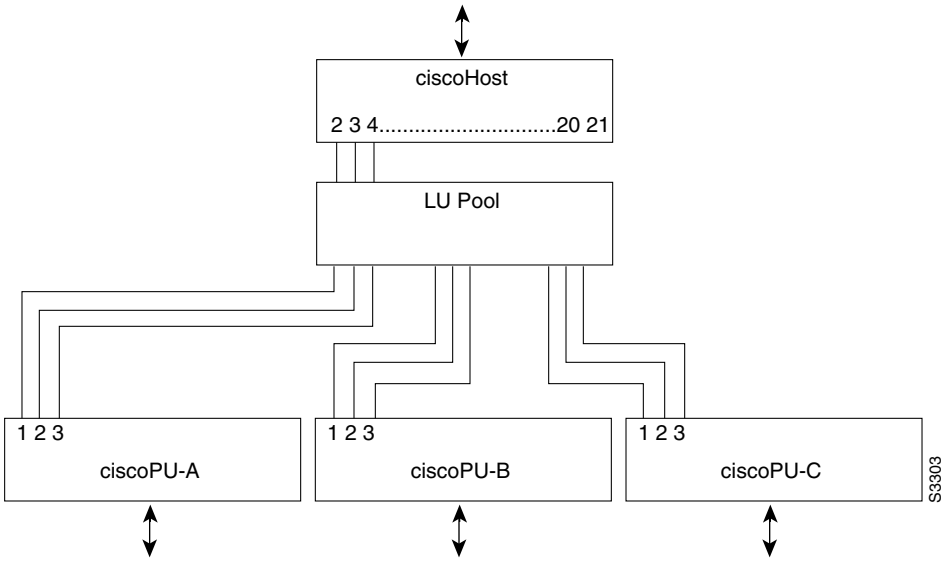
The following is a configuration file for the dedicated LU routing shown in Figure 220:

```
dspu host ciscohost xid-snd 06500001 rmac 4000.3745.0001
dspu pu ciscopu-a xid-rcv 05D00001 rmac 1000.5AED.0001
dspu lu 1 2 host ciscohost 2
dspu lu 3 3 host ciscohost 20
dspu pu ciscopu-b xid-rcv 05D00002 rmac 1000.5AED.0002
dspu lu 1 2 host ciscohost 4
dspu lu 3 3 host ciscohost 21
```

Pooled LU Routing Example

Figure 221 illustrates the use of pooled LU routing. Each upstream LU is configured in the LU pool and each downstream LU is configured as a pooled LU.

Figure 221 Pooled LU Routing



The following is a configuration file for the pooled LU routing shown in Figure 221:

```
dspu host ciscohost xid-snd 06500001 rmac 4000.3745.0001
dspu pool lupool host ciscohost lu 2 21
dspu pu ciscopu-a xid-rcv 05D00001 rmac 1000.5AED.0001
dspu lu 1 3 pool lupool
dspu pu ciscopu-b xid-rcv 05D00002 rmac 1000.5AED.0002
dspu lu 1 3 pool lupool
dspu pu ciscopu-c xid-rcv 05D00003 rmac 1000.5AED.0003
dspu lu 1 3 pool lupool
```

Upstream Host via RSRB DSPU Configuration Example

The following configuration example represents one possible definition for the network topology shown in Figure 220. This example demonstrates the configuration of an upstream host via RSRB (with local acknowledgment) and downstream PUs via Token Ring.

```
source-bridge ring-group 99
source-bridge remote-peer 99 tcp 150.10.13.1
source-bridge remote-peer 99 tcp 150.10.13.2 local-ack

dspu rsrb 88 1 99 4000.ffff.0001
dspu rsrb enable-host lsap 4

dspu host ciscohost xid-snd 06500001 rmac 4000.3172.0001 rsap 4 lsap 4
dspu pool ciscopool host ciscohost lu 2 8
dspu rsrb start ciscohost

dspu pu ciscopu1 xid-rcv 05d00001
dspu lu 2 3 pool ciscopool
```

```
dspu pu ciscopu2 xid-rcv 05d00002
dspu lu 2 4 pool ciscopool

dspu pu ciscopu3 xid-rcv 05d00003
dspu lu 2 2 pool ciscopool

dspu pu ciscopu4 xid-rcv 05d00004
dspu lu 2 2 pool ciscopool
dspu lu 3 3 host ciscohost 9

interface tokenring 0
 description tokenring connection for downstream PUs
 ring-speed 16
 dspu enable-pu lsap 8
```

DSPU over DLSw+ using Virtual Data-Link Control Configuration Example

The following example illustrates pooled LU routing over DLSw+ using virtual data-link control:

```
source-bridge ring-group 99
dlsw local-peer peer-id 150.10.16.2
dlsw remote-peer 0 tcp 150.10.16.1
!
dspu vdlc 99 4000.4500.01f0
dspu vdlc enable-pu lsap 8
dspu vdlc enable-host lsap 12
!
dspu host HOST-B xid-snd 065bbbb0 rmac 4000.7000.01f1 rsap 4 lsap 12 focalpoint
dspu pool pool-b host HOST-B lu 2 254
!
dspu pu PU3K-A xid-rcv 05d0000a rmac 4000.3000.0100 rsap 10 lsap 8
dspu lu 2 254 pool pool-b
!
dspu default-pu
dspu lu 2 5 pool pool3k-a
!
dspu vdlc start HOST-B
dspu vdlc start PU3K-A
!
interface serial 3
 description IP connection to dspu7k
 ip address 150.10.16.2 255.255.255.0
 clockrate 4000000
```

Downstream PU via SDLC DSPU Configuration Example

The following example demonstrates the configuration of downstream PUs via SDLC and an upstream host via Token Ring:

```
dspu host ciscohost xid-snd 06500001 rmac 4000.3172.0001 rsap 4 lsap 12
dspu pool ciscopool host ciscohost lu 2 11
!
dspu pu pu-sdlc0 sdlc C1 interface serial 0
dspu lu 2 6 pool ciscopool
!
dspu pu pu-sdlc1 sdlc C1 interface serial 1
dspu lu 2 6 pool ciscopool
!
```

```
interface serial 0
 description SDLC connection for pu-sdlc0
 encapsulation sdlc
 sdlc role primary
 sdlc address C1
 dspu enable-pu sdlc C1
 clockrate 56000
!
interface serial 1
 description SDLC connection for pu-sdlc1
 encapsulation sdlc
 sdlc role primary
 sdlc address C1
 dspu enable-pu sdlc C1
 clockrate 56000
!
interface tokenring 0
 description tokenring connection for ciscohost
 ring-speed 16
 dspu enable-host lsap 12
 dspu start ciscohost
```

Upstream Host via SDLC DSPU Configuration Example

The following example demonstrates the configuration of an upstream host via SDLC and downstream PUs via Token Ring and Ethernet:

```
dspu host ciscohost xid-snd 06500001 sdlc C1 interface serial 0
dspu pool ciscopool host ciscohost lu 2 11
!
dspu pu pu-token rmac 4000.4444.0001 rsap 4 lsap 8
dspu lu 2 6 pool ciscopool
!
dspu pu pu-ether rmac 0200.2222.0001 rsap 4 lsap 8
dspu lu 2 6 pool ciscopool
!
interface serial 0
 description SDLC connection for ciscohost
 encapsulation sdlc
 sdlc role secondary
 sdlc address C1
 dspu enable-host sdlc C1
 clockrate 56000
 dspu start ciscohost
!
interface tokenring 0
 description tokenring connection for pu-token
 ring-speed 16
 dspu enable-pu lsap 8
!
interface ethernet 0
 description Ethernet connection for pu-ether
 dspu enable-pu lsap 8
```

Downstream PU via QLLC/X.25 DSPU Configuration Example

The following example demonstrates the configuration of a downstream PU via QLLC/X.25 and upstream host via Ethernet:

```
x25 routing
!
dspu host ciscohost xid-snd 06500001 rmac 0200.2222.0001 rsap 4 lsap 12
dspu pool ciscopool host ciscohost lu 2 11
!
dspu pu pu-qllc x25 320108 qllc 08
dspu lu 2 11 pool ciscopool
!
interface serial 0
 description QLLC connection for pu-qllc
 encapsulation x25
 x25 address 3202
 x25 map qllc 320108
 dspu enable-pu qllc 8
!
interface ethernet 0
 description Ethernet connection for pu-ether
 dspu enable-host lsap 12
 dspu start ciscohost
!
 x25 route ^3202.* alias serial 0
```

Upstream Host via Frame Relay DSPU Configuration Example

The following example demonstrates the configuration of an upstream host via Frame Relay and downstream PUs via Token Ring and Ethernet:

```
dspu host ciscohost xid-snd 06500001 dlci 200 rsap 4 lsap 12
dspu pool ciscopool host ciscohost lu 2 11
!
dspu pu pu-token rmac 4000.4444.0001 rsap 4 lsap 8
dspu lu 2 6 pool ciscopool
!
dspu pu pu-ether rmac 0200.2222.0001 rsap 4 lsap 8
dspu lu 2 6 pool ciscopool
!
interface serial 0
 description Frame Relay connection for ciscohost
 encapsulation frame-relay ietf
 frame-relay map llc2 200
 dspu enable-host lsap 12
 dspu start ciscohost
!
interface tokenring 0
 description tokenring connection for pu-token
 ring-speed 16
 dspu enable-pu lsap 8
!
interface ethernet 0
 description Ethernet connection for pu-ether
 dspu enable-pu lsap 8
```

DSPU NCIA Configuration Example

The following example illustrates an NCIA client/server session using DSPU:

```
ncia server 1 10.2.20.4 4000.3745.0001 1000.0000.0001 128
!
dspu ncia 1
dspu ncia enable-pu lsap 8
!
dspu host HOST-9370 xid-snd 11100001 rmac 4000.1060.1000 rsap 4 lsap 4
!
dspu pu CISCOPU-A xid-rcv 01700001
dspu lu 2 6 host HOST-9370 2
!
interface TokenRing 0
 ring-speed 16
 llc2 xid-retry-time 0
 dspu enable-host lsap 4
 dspu start HOST-9370
!
```

SNA Service Point Support Configuration Example

The following is an example of an RSRB configuration that implements SNA Service Point:

```
source-bridge ring-group 99
source-bridge remote-peer 99 tcp 150.10.13.2 local-ack
!
sna rsrb 88 1 99 4000.ffff.0001
!
sna host CNM02 xid-snd 05dbc000 rmac 4001.3745.1088 rsap 4 lsap 4 focalpoint
sna rsrb enable-host lsap 4
sna rsrb start CNM02
!
```

SNA Service Point over DLSw+ Using Virtual Data-Link Control Configuration Example

The following is an example of an SNA Service Point configuration that uses virtual data-link control over DLSw+:

```
source-bridge ring-group 99
dlsw local-peer peer-id 150.10.16.2
dlsw remote-peer 0 tcp 150.10.16.1
!
sna vdlc 99 4000.4500.01f0
sna vdlc enable-host lsap 12
!
sna host HOST-B xid-snd 065bbbb0 rmac 4000.7000.01f1 rsap 4 lsap 12 focalpoint
!
sna vdlc start HOST-B
!
interface serial 3
 description IP connection to dspu7k
 ip address 150.10.16.2 255.255.255.0
 clockrate 4000000
```

Configuring SNA Switching Services

This chapter describes SNA Switching Services (SNASw), which supersedes all functionality previously available in the Advanced Peer-to-Peer Networking (APPN) feature in the Cisco IOS software. SNASw configuration will not accept the previous APPN configuration commands. Previous APPN users should use this chapter to configure APPN functionality using the new SNASw commands.

For a complete description of the SNASw commands mentioned in this chapter, refer to the "SNA Switching Services Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page lv in the "Using Cisco IOS Software" chapter.

Technical Overview

SNASw provides an easier way to design and implement networks with Systems Network Architecture (SNA) routing requirements. Previously, this network design was accomplished using APPN with full network node (NN) support in the Cisco router. This type of support provided the SNA routing functionality needed, but was inconsistent with the trends in Enterprise networks today. The corporate intranet is replacing the SNA WAN. Enterprises are replacing their traditional SNA network with an IP infrastructure that supports traffic from a variety of clients, using a variety of protocols, requiring access to applications on a variety of platforms, including SNA applications on enterprise servers.

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

While SNA routing is still required when multiple servers must be accessed, the number of nodes required to perform this function is decreasing as the IP infrastructure grows and as the amount of native SNA traffic in the network decreases.

SNASw enables an enterprise to develop their IP infrastructure, while meeting SNA routing requirements.

The number of NNs in the network and the amount of broadcast traffic are reduced. Configuration is simplified, and SNA data traffic can be transported within the IP infrastructure. The following features provide this functionality:

High Performance Routing (HPR)-Capable SNA Routing Services

SNASw provides the following SNA routing functions:

- Routes SNA sessions between clients and target SNA data hosts.
- Controls SNA traffic in a multiprotocol environment in conjunction with other Cisco IOS quality of service (QoS) features.
- Supports networks with a high proportion of SNA traffic and multiple enterprise servers, especially those that continue to support the traditional SNA endstation platform and new client types.
- Supports all types of SNA application traffic including traditional 3270 and peer LU 6.2.
- Supports an OS/390 Parallel Sysplex configuration, working in conjunction with the IBM Communications Server for S/390 (formerly VTAM) and the MVS Workload Manager, to provide higher availability in the data center using the HPR feature.
- Supports System Services Control Point (SSCP) services to downstream SNA devices using the Dependent LU Requester (DLUR) feature.
- Provides dynamic link connectivity with or without connection networks (CNs), which eliminates much of the configuration required in networks with numerous data hosts.

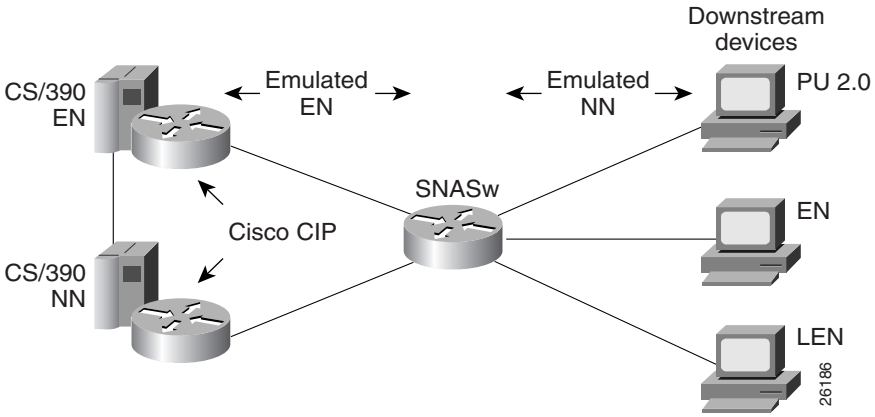
Branch Extender

The Branch Extender (BEX) function enhances scalability and reliability of SNA routing nodes by eliminating topology updates and broadcast directory storms that can cause network instability. BEX appears as an NN to downstream end node (EN), low-entry networking (LEN) node, and PU 2.0 devices, while also appearing as an EN to upstream devices. The BEX function eliminates APPN topology and APPN broadcast search flows between SNASw nodes and the SNA data hosts in the network. This feature is key to providing a reliable turn-key installation because the network administrator no longer needs to develop in-depth knowledge of the level and characteristics of broadcast directory search and topology update traffic in the network. Such knowledge and analysis was commonly required to build successful networks utilizing NN technology without BEX.

SNASw enables BEX functionality by default. SNASw treats all defined links as BEX “uplinks” and all dynamic links created by stations connecting into SNASw as Branch Extender “downlinks.” No specific configuration is necessary to enable BEX functionality.

Figure 222 illustrates the BEX functionality.

Figure 222 BEX Functionality

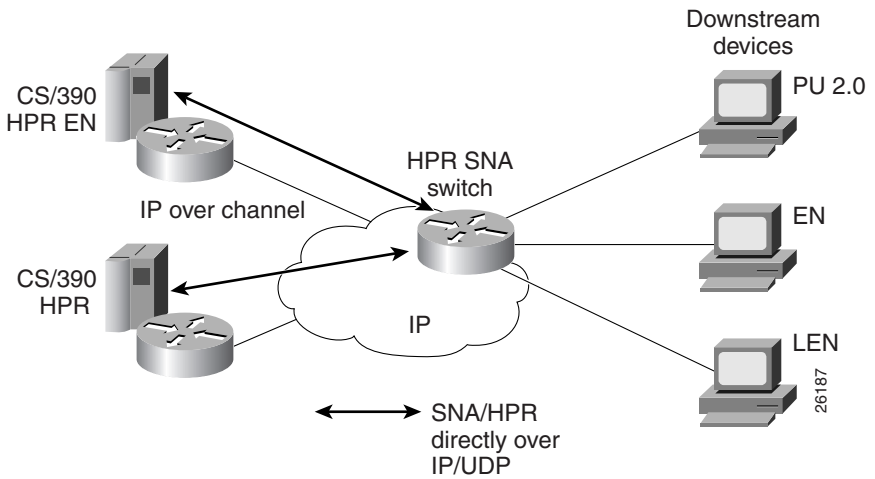


Enterprise Extender (HPR/IP)

SNASw also supports the Enterprise Extender (EE) function. EE offers SNA HPR support directly over IP networks. EE also uses connectionless User Datagram Protocol (UDP) transport. SNA COS and transmission priority are maintained by mapping the transmission priority to the IP precedence and by mapping transmission priority to separate UDP port numbers, allowing the IP network to be configured based on these elements. Cisco's IP prioritization technologies, such as weighted fair queueing (WFQ), prioritize the traffic through the IP network. EE support on the IBM Communications Server for S/390 allows users to build highly reliable SNA routed networks that run natively over an IP infrastructure directly to the Enterprise servers. These network designs reduce points of failure in the network and provide reliable SNA networks.

Figure 223 illustrates the EE functionality.

Figure 223 EE Functionality



Usability Features

SNASw contains the following usability features designed to make SNA networks easier to design and maintain:

Dynamic CP Name Generation Support

When scaling the SNASw function to hundreds or thousands of nodes, many network administrators find that defining a unique control point (CP) name on each node generates unnecessary configuration overhead. Dynamic CP name generation offers the ability to use the Cisco IOS hostname as the SNA CP name or to generate a CP name from an IP address. These facilities reuse one SNASw configuration across many routers and eliminate the specific configuration coordination previously required to configure a unique CP name for each SNA node in the network. Administrators can still explicitly configure the CPname within the SNASw configuration.

Dynamic SNA BTU Size

SNASw analyzes the maximum transmission unit (MTU) size of router interfaces configured for native LAN interfaces such as Token Ring, Ethernet and FDDI, and dynamically assign the best MTU values for that specific port. For other interface types, SNASw provides the maxbtu parameter on the port statement. For served dependent PU 2.0 devices, SNASw uses the downstream MAXDATA value from the host and then dynamically sets the SNA BTU for that device to the MAXDATA value.

DLUR Connect-Out

SNASw can receive connect-out instructions from the IBM Communications Server for S/390. This function allows the system to dynamically connect-out to devices that are configured on the host with the appropriate connect-out definitions. This feature allows connectivity to SNA devices in the network that were traditionally configured for connect-out from the host.

Note DLUR connect-out can be performed over any supported data-link type.

Responsive Mode Adaptive Rate-Based Flow Control

Early HPR implementations failed to perform well in environments subject to packet loss (for example, Frame Relay, IP transport) and performed poorly when combined with other protocols in multiprotocol networks. SNASw implements the second-generation HPR flow-control architecture, called Responsive Mode Adaptive Rate-Based (ARB) architecture. Responsive Mode ARB addresses all the drawbacks of the earlier ARB implementation, providing faster ramp-up, better tolerance of lost frames, and better tolerance of multiprotocol traffic.

User-Settable Port Limits

SNASw offers full control over the number of devices supported by a specific port. The max-links configuration on the SNASw port controls the number of devices that are served by this port. When the max-links limit is reached, SNASw no longer responds to test frames attempting to establish new connections. SNASw allows load sharing among different SNASw nodes that offer service to the same SNA MAC addresses.

Management Enhancements

SNASw contains the following enhanced tools for managing SNA networks:

- Console Message Archiving, page 6
- Data-Link Tracing, page 6
- Interprocess Signal Tracing, page 6
- MIB Support for Advanced Network Management Awareness, page 6

Console Message Archiving

Messages issued by SNASw are archived in a buffer log that is queried and searched at the console or transferred to a file server for analysis. Each message has a single line that identifies the nature of the event that occurred. The buffer log also maintains more detailed information about the message issued.

Data-Link Tracing

SNA frames entering or leaving SNASw are traced to the console or to a cyclic buffer. These frames are analyzed at the router or transferred to a file server for analysis. The trace is sent to a file server in a SNA-formatted text file or in binary format readable by existing traffic analysis applications.

Interprocess Signal Tracing

The SNASw internal information is traced in binary form, offering valuable detailed internal information to Cisco support personnel. This information helps diagnose suspected defects in SNASw.

MIB Support for Advanced Network Management Awareness

SNASw supports the following Management Information Bases (MIBs):

- IETF draft standard DLUR MIB (RFC 2232), which defines objects for monitoring and controlling network devices with DLUR (Dependent LU Requester) capabilities.
- IETF draft standard APPN MIB (RFC 2455), which defines objects for monitoring and controlling network devices with Advanced Peer-to-Peer Networking (APPN) capabilities.
- APPN Traps MIB (RFC 2456), which defines objects for receiving notifications from network devices with APPN and DLUR capabilities. This MIB proactively send traps with information about changes in SNA resource status. This implementation reduces the frequency of SNMP polling necessary to manage SNA devices in the network.

The CiscoWorks Blue Maps application retrieves relevant SNASw data from these MIBs and displays it in a manner that simplifies and speeds up problem isolation and resolution.

LAN and IP-Focused Connection Types

SNASw supports several connection types to serve all SNA connectivity options, including the following types:

Token Ring, Ethernet, and FDDI

SNASw natively supports connectivity to Token Ring, Ethernet, and FDDI networks. In this configuration mode, the MAC address used by SNASw is the locally configured or default MAC address of the interface.

Virtual Token Ring

Using virtual Token Ring allows SNASw access to a source-route bridging (SRB) network, which allows the following configuration:

Attachment to Local LANs

Virtual Token Ring allows you to connect to local LAN media through SRB technology. Virtual Token Ring and SRB allow SNASw to respond to multiple MAC addresses over the same physical interface. Because there is no limit to the number of virtual Token Ring interfaces that can connect to a specific LAN, you can configure multiple MAC addresses, which respond to SNA requests over the same LAN. When using native LAN support, SNASw responds only to requests that target the MAC address configured on the local interface.

Connection to Frame Relay Transport Technologies

Virtual Token Ring and SRB connect SNASw to a SNA Frame Relay infrastructure. FRAS host and SRB Frame Relay are configured to connect virtual Token Ring interfaces that offer SNASw support for Frame Relay boundary access node (BAN) or boundary network node (BNN) technology.

Connection to Channel Interface Processor and Channel Port Adapter

Virtual Token Ring and SRB can be used to connect SNASw to the Channel Interface Processor (CIP) or Channel Port Adapter (CPA) in routers that support those interfaces.

Virtual Data-Link Control

SNASw uses Virtual Data-Link Control (VDLC) to connect to data-link switching plus (DLSw+) transport and local switching technologies. VDLC is used for a number of connectivity options, including the following two:

- Transport over DLSw+ Supported Media, page 8
- DLC Switching Support for Access to SDLC and QLLC, page 8

Transport over DLSw+ Supported Media

Using VDLC, SNASw gains full access to the DLSw+ transport facilities, including DLSw+ transport over IP networks, DLSw+ transport over direct interfaces, and DLSw+ support of direct Frame Relay encapsulation (without using IP).

DLC Switching Support for Access to SDLC and QLLC

Through VDLC, SNASw gains access to devices connecting through synchronous data link control (SDLC) and qualified logical link control (QLLC). This access allows devices connecting through SDLC and QLLC access to SNASw.

Native IP Data-Link Control (HPR/IP)

SNASw support for the EE function provides direct HPR over UDP connectivity. This support is configured for any interface that has a configured IP address. HPR/IP uses the interface IP address as the source address for IP traffic originating from this node. An enhancement was introduced in Cisco IOS Release 12.3(14)T, allowing EE to work with IP v6 by associating a host name with the port/link. Host names can also be used for IPv4 ports/links to resolve issues with NAT and connection network.

Benefits of SNASw

SNASw provides the following benefits:

- Scalable APPN Networks, page 8
- IP Infrastructure Support, page 9
- Reduced Configuration Requirements, page 9
- Network Design Simplicity, page 9
- Improved Availability, page 9
- Increased Management Capabilities, page 9
- Architectural Compliance, page 9

Scalable APPN Networks

With the BEX function, the number of network nodes and the amount of broadcast traffic are reduced.

IP Infrastructure Support

Limiting SNASw routers to the data center and using the BEX function eliminates SNA broadcasts from the IP network. With EE, SNA traffic is routed using the IP routing infrastructure while maintaining end-to-end SNA services.

Reduced Configuration Requirements

By eliminating NNs and using the BEX function, configuration tasks are minimized. Additionally, Cisco has enhanced its auto-configuration capability to eliminate previously required commands.

Network Design Simplicity

By placing all SNA routers in the data center, fewer SNA routers are required, and they can be easily configured using virtually identical configurations.

Improved Availability

By adding Cisco-unique capabilities to connect-out and distribute traffic across multiple ports, access to resources is improved. Additionally, by supporting the newest HPR ARB flow control algorithm, bandwidth management for SNA traffic is improved.

Increased Management Capabilities

Two new traces, interprocess and data-link, provide an easier way to view SNASw activity. The APPN Trap MIB allows the user to notify the operator in event of a debilitating problem. Console message archiving provides better tracking of network activity. The ability to format traces so that they are readable by other management products simplifies network management because results are more readily available.

Architectural Compliance

SNASw interfaces with SNA implementations on the market: upstream NNs, ENs, LENs and PU 2.0. It also provides full DLUR support to allow dependent PU and LU traffic to flow over the APPN network to SNA data hosts.

SNASw Configuration Task List

To configure SNASw in your network, perform the tasks discussed in the following sections. Because of the hierarchical nature of SNASw definitions, configure SNASw in the order specified. Definition of an SNASw CP name and at least one SNASw port are required. The other tasks are optional. Depending on your network, the optional tasks might need to be performed.

- Defining an SNASw Control Point Name, page 10 (Required)
- Configuring a DLUS, page 10 (Optional)
- Configuring DLC Support, page 11 (Optional)
- Configuring Hostnames for EE, page 11 (Optional)

- Defining an SNASw Port, page 12 (Required)
- Defining an SNASw Link, page 13 (Optional)
- Defining an SNASw Partner LU Location, page 13 (Optional)
- Starting SNASw and SNASw Ports and Links, page 14 (Optional)
- Stopping SNASw and SNASw Ports and Links, page 14 (Optional)

See the "SNASw Configuration Examples" section on page 16 for examples.

Defining an SNASw Control Point Name

An SNASw CP definition is required to use SNASw. This definition adds the fully qualified CP name for the node. The fully qualified CP name for the node is a combination of a network identifier and a CP name. The network identifier is typically configured to match the identifier configured in the SNA hosts in the network. The CP name identifies this node uniquely within the particular subnetwork.

To define an SNASw CP name, use the following command in global configuration mode:

Command	Purpose
Router# **snasw cpname** {*netid.cpname* | *netid* [*hostname* | **ip-address** *interface-name*]} [**hung-pu-awareness** *timer-value*] [**hung-session-awareness** *timer-value*] [**locate-timeout** *timeout-value*] [**max-pacing-window** *max-value*] [**remove-rscvs**] [**station-segmentation**]	Defines an SNASw CP name.

Note Configuring a CP name activates SNASw. Conversely, removing a CP name definition deactivates it.

Configuring a DLUS

If you plan to provide services to dependent LUs connecting to this SNASw node, you will be using the DLUR functionality within SNASw. SNASw defaults to using its current active upstream Network Node Server (NNS) as the preferred Dependent LU Server (DLUS) for the node. To override this default and explicitly configure the DLUS name, configure the **snasw dlus** command. In addition, you can configure node-wide defaults for the DLUS and backup DLUS that this node contacts.

To specify DLUR or DLUS services for this CP name, use the following command in SNASw control point configuration mode:

Command	Purpose
Router# **snasw dlus** *primary-dlus-name* [**backup** *backup-dlus-name*] [**prefer-active**] [**retry** *interval count*] [**once**]	Specifies the parameters related to DLUR/DLUS functionality.

Configuring DLC Support

There are several ways that SNASw enables connectivity over different interface types. In the simplest cases, using automatically configured real LAN interfaces enables default interface definitions. SNASw is also capable of connecting to virtual interfaces that are not preconfigured on the router.

Virtual Token Ring interfaces are useful for connections to a CIP/CPA in the same router and for connectivity to Frame Relay transport solutions via SRB. Multiple virtual Token Ring interfaces allow SNASw to respond to multiple MAC addresses through the same real router LAN interface. Use the following commands to configure a virtual interface:

	Command	Purpose
Step 1	`Router# interface virtual-tokenring number`	Configures a virtual Token Ring interface to connect to an SRB infrastructure.
Step 2	`Router# source-bridge vring bridge ring-group`	Associates a virtual Token Ring interface with a source-route bridge group.
Step 3	`Router# source-bridge spanning`	Indicates this interface should respond to spanning-tree explorers.
Step 4	`Router# mac-address mac-address`	Configures a MAC address on a real or virtual LAN interface.

Configuring Hostnames for EE

When using IPv6 for EE, you must use hostnames rather than explicit IP addresses. When using IPv4 for EE you can optionally use hostnames rather than the explicit IP address and this allows connection network to work through a NAT boundary.

When EE host name support is used, both local and remote IP addresses must be able to be resolved via host names. The host name configuration must be in place before you configure the SNASw port and link or links. Hostnames can be resolved in the local cache (through the ip host or ipv6 host statements) or by configuring a DNS.

To configure hostnames for EE you can use one of the options in global configuration mode from the following table:

Command	Purpose
Option 1	
`Router# ip hostname ip-address`	Creates a local ip host cache entry for each local and remote address.
Option 2	
`Router# ipv6 hostname ipv6-address`	Creates a local ipv6 host cache entry for each local and remote address.
Option 3	
`Router# ip name-server ip-address`	Identifies the DNS for hostname resolution.

Defining an SNASw Port

An SNASw port definition associates SNA capabilities with a specific interface that SNASw will use. Each interface that is used for SNASw communications requires an SNASw port definition statement.

A port can also be associated with the VDLC or HPR/IP features. The VDLC feature enables SNASw to send and receive traffic to other Cisco IOS software features such as DLSw+. If a port is associated with a VDLC interface, that port does not take an interface name as generally required by the **snasw port** command.

The HPR/IP feature establishes SNASw links over IP networks. If a port is associated with an HPR/IP interface, then you must configure the **hpr-ip** keyword first, followed by the interface name.

To associate a port with a specific interface, use the following commands beginning in global configuration mode:

Command	Purpose
Router# **snasw port** *portname* **hpr-ip** *interface-name* [**hostname** *v4-or-v6-hostname*][**ldlc** [*liveness-time t1-retry-time t1-retry-count*]] [**maxbtu** *max-btu-size*][**vnname** *virtual-node-name*][**no-limres**][**no-start**]	Specifies the IP DLC used by SNASw for hpr-ip ports.
Router# **snasw port** *portname* {**vdlc** *ring-group* **mac** *mac-address* |**Virtual-TokenRing** *interface-name*}[**conntype nohpr** | **len** | **dyncplen** | **dialoutlen**][**hpr-sap** *hpr-sap-value*][**max-links** *link-limit-value*][**maxbtu** *max-btu-size*][**nns-required**][**sap** *sap-value*][**vnname** *virtual-node-name*][**no-limres**]][**no-start**]	Specifies the Virtual DLC used by SNASw for Virtual DLC and Virtual Token-Ring ports.
Router# **snasw port** *portname interface-name* [**conntype nohpr** | **len** | **dyncplen**| **dialoutlen**][**hpr-sap** *hpr-sap-value*][**max-links** *link-limit-value*][**maxbtu** *max-btu-size*][**sap** *sap-value*][**vnname** *virtual-node-name*][**no-limres**]][**no-start**]	Specifies the DLCs used by SNASw for all other ports.

Note SNASw ports do not dynamically adjust to interface configuration changes that are made when SNASw is active. For example, if you change an interface MAC address or MTU, SNASw may not recognize the new value. If you want to make changes to an interface and want SNASw to adjust to the new interface changes, you may need to either delete and redefine the port that is using that interface or stop and restart SNASw.

The interface must be defined before the ports that use them are defined and activated.

SNASw does not support EtherChannel interfaces (neither port-channel interfaces nor Fast Ethernet interfaces configured with the **channel-group** command). Do not try to configure a SNASw port with either of these EtherChannel interface types.

Caution Changing active SNASw interfaces might interrupt SNASw connections.

Defining an SNASw Link

In many cases, if the destination LU is initiating the connection, a link definition is not necessary. A link definition is built dynamically when the destination LU initiates the connection. Links typically need to be defined for upstream connectivity. Downstream devices initiate connectivity into SNASw; therefore, links should not be defined on SNASw to downstream devices.

In SNASw link configuration, you must associate the link with the SNASw port that it will use. For all traditional links, the **snasw link** command must be associated with a remote MAC address. The MAC address identifies the partner address to which SNASw attempts to establish a link. For all HPR/IP links, the command is associated with a remote IP address. The IP address identifies the partner address to which SNASw attempts to establish a link.

To define an SNASw logical link, use the following command in global configuration mode:

Command	Purpose
`Router# snasw link linkname port portname [rmac mac-address | ip-dest ip-address] [rsap sap-value] [nns] [tgp [high | low | medium]] [nostart]`	Defines an SNASw logical link.
`Router# snasw link linkname port portname [rmac mac-address | host-dest v4-or-v6-hostname | ip-dest ip-address][rsap sap-value] [nns] [tgp [high | low | medium]] [nostart]`	Configures upstream links.

Defining an SNASw Partner LU Location

The SNASw directory stores names of resources and their owners. Usually this information is learned dynamically using Locate searches. You might want to manually define the location of specific resources. SNASw is known for its dynamic capabilities, not its need for system definition. For this reason, and for easier management, define location names only when necessary.

When a LEN node connects into an SNASw node, SNASw dynamically learns the CP name of the LEN and places it in its directory. In addition, SNASw dynamically learns the LU names of all LUs on the LEN that initiate independent sessions. Only define the location when an ILU on a LEN device is not sharing the node's CP name and does not initiate the first session. In all other cases the LU's location will be learned dynamically.

The directory entry is created the next time the LEN node connects in. If there is already a link to the LEN node active and you add a new snasw location statement, it will not take effect until the next time the LEN CP connects in.

To define a resource location, use the following command in global configuration mode:

Command	Purpose
`Router# snasw location resource-name owning-cp cpname`	Configures the location of a resource.

Note You must configure an owning CP for each partner LU configured. The owning CP is the CP name for the LEN node on which the partner resource resides. Location definitions are never required for resources located on APPN ENs or NNs.

Starting SNASw and SNASw Ports and Links

Unless otherwise defined with the **nostart** operand, SNASw starts automatically when a CP name is configured, and SNASw ports and links are also automatically started once they are configured. If stopped, they can be restarted using one of the following privileged EXEC commands, as needed:

Command	Purpose
`Router# snasw start`	Starts SNASw.
`Router# snasw start link linkname`	Activates the specified SNASw link.
`Router# snasw start port portname`	Activates the specified SNASw port.

Stopping SNASw and SNASw Ports and Links

Unless otherwise defined with the **nostart** operand, SNASw and SNASw port and link definitions are started automatically when SNASw starts. To stop SNASw or to stop SNASw ports and links when making configuration changes or when resetting the ports or links, use one of the following commands in privileged EXEC mode, as needed:

Command	Purpose
`Router# snasw stop`	Deactivates SNASw.
`Router# snasw stop link linkname`	Deactivates the specified SNASw link.
`Router# snasw stop port portname`	Deactivates the specified SNASw port.

Note Removing a CP name definition stops SNASw.

Verifying SNASw

To verify that you have connectivity between SNASw and other nodes supporting APINGD transaction program, issue the **ping sna** command. To start an independent LU-LU session and send simple APINGD test data traffic, also issue the **ping sna** command.

Monitoring and Maintaining SNASw

You can monitor the status and configuration of SNASw by issuing any of the following commands in privileged EXEC mode:

Command	Purpose
Router# **ping sna** [*-1*] [**-c** *consecutive packets*] [**-i** *number-iterations*] [**-m** *mode*] [**-n**] [**-r**] [**-s** *size*][**-t** tpname] [**-u** *userid* **-p** *password*] destination	Initiates an APPC session with a named destination LU to run the APING transaction program to check network integrity and timing characteristics.
Router# **show snasw class-of-service** [**brief** | **detail**]	Displays the predefined COS definitions.
Router# **show snasw connection-network** [**brief** | **detail**]	Displays the connection networks (virtual nodes) defined to the local node.
Router# **show snasw directory** [**name** *resourcenamefilter*] [**brief** | **detail**]	Displays the SNASw directory entries.
Router# **show snasw dlus** [**brief** | **detail**]	Displays the SNASw DLUS objects.
Router# **show snasw link** [**brief** | **detail**] [**cpname** *cpnamefilter*] [**name** *linknamefilter*] [**port** *portnamefilter*] [**rmac** *macfilter*] [**xid** *xidfilter*]	Displays the SNASw link objects.
Router# **show snasw lu** [**brief** | **detail**][**name** *luname*] [**pu** *puname*]	Displays the SNASw dependent LUs.
Router# **show snasw mode**	Displays the SNASw modes.
Router# **show snasw node**	Displays details of the SNASw operation.
Router# **show snasw port** [**brief** | **detail**] [**name** *portnamefilter*]	Displays the SNASw port objects.
Router# **show snasw pu** [**brief** | **detail**] [**dlus** *dlusfilter*] [**name** *punamefilter*]	Displays the SNASw PUs.
Router# **show snasw rtp** [**brief** | **detail**] [**class-of-service** *cosname*] [**cpname** *netid.cpname*] [**name** *connectionnamefilter*] [**tcid** *tcidconnection*]	Displays the SNASw RTP connections.

Troubleshooting Tips

You can troubleshoot SNASw by issuing any of the following commands in privileged EXEC mode:

Command	Purpose
Router# **ping sna** [*-1*] [**-c** *consecutive packets*] [**-i** *number-iterations*] [**-m** *mode*] [**-n**] [**-r**] [**-s** *size*][**-t** tpname] [**-u** *userid* **-p** *password*] destination	Initiates an APPC session with a named destination LU to run the APING transaction program to check network integrity and timing characteristics.
Router# **show snasw dlctrace** [**all** | **last** | **next**] [**brief** | **detail**] [**filter** *filter-string*] [**id** *recordid*]	Displays the captured DLC trace information to the console.
Router# **show snasw ipstrace** [**all** | **next** | **last**] [**filter** *filterstring*] [**id** *recordid*]	Displays interprocess signal trace on the router console.
Router# **show snasw pdlog** [**brief** | **detail**] [**all**] [**last**] [**next**] [**filter** *filterstring*] [**id** *recordid*]	Displays entries in the cyclical problem determination log to the console.
Router# **show snasw summary-ipstrace** [**id** *recordid*] [**last** *number-records* | **filter** *number-records* | **all** | **next** | **last**]	Displays the special “footprint” summary interprocess signal trace on the router console.
Router# **snasw dump**	Initiates file transfer of SNASw trace files from internal buffers to a file server.

Command	Purpose
Router# **snasw pathswitch** [*rtp-connection-name* | **all**]	Forces an HPR pathswitch for an RTP connection.
Router# **snasw start arbdata** *local-tcid*	Starts the writing of debug messages to the router log containing adaptive rate based (ARB) algorithm data for a particular Rapid Transport Protocol (RTP) connection specified by the *local-tcid*. • This is useful for tracking down RTP performance problems, but it can generate a very large number of log entries. • For this reason it should only be started when you have already configured the **no logging console debug** and the **logging buffered debug** commands.
Router# **snasw stop arbdata** *local-tcid*	Stops the writing of debug messages to the router log containing ARB algorithm data for a particular RTP connection specified by the *local-tcid*.

You can also troubleshoot SNASw by issuing any of the following commands in global configuration mode:

Command	Purpose
Router# **snasw dlcfilter** [**link** *linkname* [**session** *session-address*]] [**port** *portname*] [**rmac** *mac-address-value* [**session** *session-address*]] [**rtp** *rtpname* [**session** *session-address*]] [[**type** [**cls**] [**hpr-cntl**] [**hpr-data**] [**isr**] [**xid**]]	Filters frames captured by the **snasw dlctrace** or **debug snasw dlc** commands.
Router# **snasw dlctrace** [**buffer-size** *buffer-size-value*] [**file** *filename*] [**frame-size** *frame-size-value*] [**format brief** | **detail** | **analyzer**] [**nostart**]	Traces frames arriving at and leaving SNASw.
Router# **snasw event** [**cpcp**] [**dlc**] [**implicit-ls**] [**port**]	Indicates which events are logged to the console.
Router# **snasw ipsfilter** [**as**] [**asm**] [**bm**] [**ch**] [**cpc**] [**cs**] [**di**] [**dlc**] [**dma**] [**dr**] [**ds**] [**es**] [**ha**] [**hpr**] [**hs**] [**lm**] [**mds**] [**ms**] [**nof**] [**pc**] [**ps**] [**pu**] [**px**] [**rm**] [**rtp**] [**ru**] [**scm**] [**sco**] [**sm**] [**spc**] [**ss**] [**trs**]	Filters interprocess signal trace elements being traced via the **snasw ipstrace** or **debug snasw ips** commands.
Router# **snasw ipstrace** [**buffer-size** *buffer-size-value*] [**file** *filename*]	Sets up a trace buffer and begins tracing IPS trace elements.
Router# **snasw pdlog** [*problem* | *error* | *info*] [**buffer-size** *buffer-size-value*] [**file** *filename*]	Controls logging of messages to the console and the SNA problem determination log cyclic buffer.

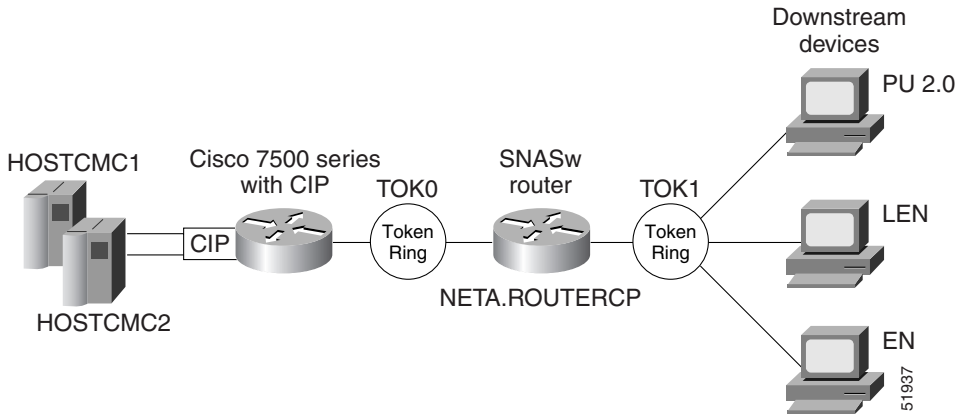
SNASw Configuration Examples

This section provides the following configuration examples:

SNASw over Token Ring without HPR Configuration Example

Figure 224 illustrates a basic SNASw link over Token Ring without HPR. In this figure, Port TOK0 is used for upstream links toward the host, and Port TOK1 is used for downstream devices connecting to SNASw. These devices are configured to connect to 4000.1234.abcd. The conntype nohpr operand is designed to turn off HPR capabilities on upstream and downstream links.

Figure 224 SNASw over Token Ring without HPR



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

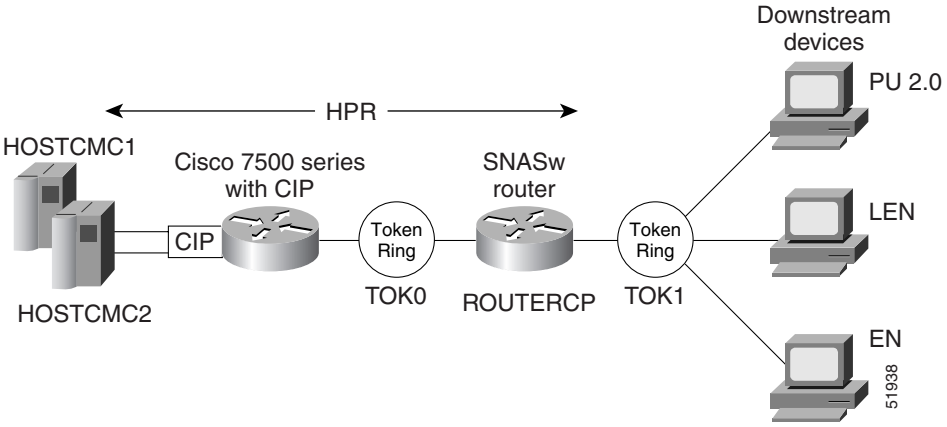
The configuration for SNASw over Token Ring without HPR is as follows:

```
interface TokenRing0/0
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ring-speed 16

interface TokenRing0/1
 mac-address 4000.1234.abcd
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ring-speed 16

snasw rtp pathswitch-timers 160 80 40 20
snasw cpname NETA.ROUTERCP
snasw dlus NETA.HOSTCMC1 backup NETA.HOSTCMC2
snasw port TOK0 TokenRing0/0 conntype nohpr
snasw port TOK1 TokenRing0/1 conntype nohpr
snasw link HOSTCMC1 port TOK0 rmac 4000.aaaa.cccc
snasw link HOSTCMC2 port TOK0 rmac 4000.aaaa.dddd
```

SNASw over Token Ring with HPR Configuration Example

Figure 225 illustrates a basic SNASw link over Token Ring with HPR support. In this figure, Port TOK0 is used for upstream links toward the host, and Port TOK1 is used for downstream devices connecting to SNASw. These devices are configured to connect to 4000.1234.abcd.

Figure 225 SNASw over Token Ring with HPR



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

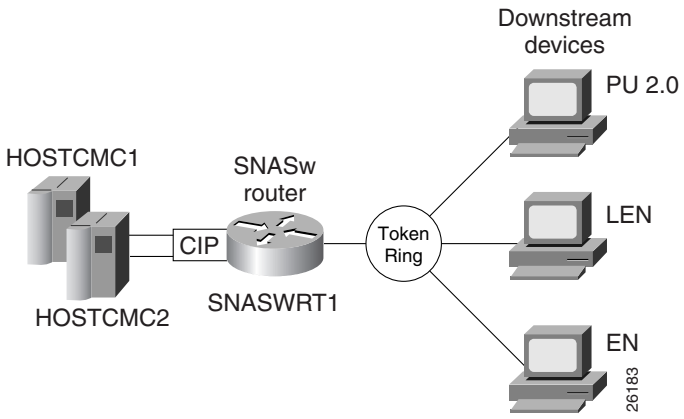
The configuration for SNASw over Token Ring allowing HPR is as follows:

```
interface TokenRing0/0
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ring-speed 16

interface TokenRing0/1
 mac-address 4000.1234.abcd
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ring-speed 16

snasw cpname NETA.ROUTERCP
snasw dlus NETA.HOSTCMC1 backup NETA.HOSTCMC2
snasw port TOK0 TokenRing0/0
snasw port TOK1 TokenRing0/1
snasw link HOSTCMC1 port TOK0 rmac 4000.aaaa.cccc
snasw link HOSTCMC2 port TOK0 rmac 4000.aaaa.dddd
```

SNASw Connecting to a CIP over Virtual Token Ring with SRB Configuration Example

In Figure 226, SNASw co-exists with CSNA CIP channel support in the same router. Two adapters are opened on the CIP, one from HOSTCMC1 on adapter 1 and one from HOSTCMC2 on adapter 2. SNASw is configured to connect these two hosts through port CIP via the SRB infrastructure. In addition, SNASw has two ports configured for downstream devices. Using this configuration, SNASw responds to downstream clients connecting to 4000.1234.1088 and 4000.1234.1089 through a single Token Ring interface (Token Ring 0/0). The router's hostname is used to derive an SNASw CP name, which is NETA.SNASWRT1.

Figure 226 SNASw Connecting to a CIP over Virtual Token Ring with SRB



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

The configuration for SNASw connecting to a CIP over virtual Token Ring with SRB is as follows:

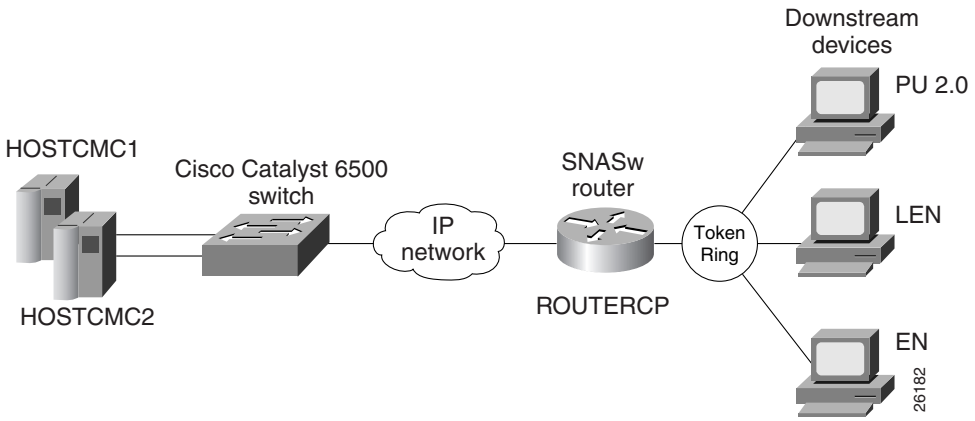
```
hostname snaswrt1
!
source-bridge ring-group 100
source-bridge ring-group 200
!
interface Channel2/1
 no ip address
 no keepalive
 csna E040 70
 csna E020 72
!
interface Channel2/2
 no ip address
 no keepalive
 lan TokenRing 0
 source-bridge 101 1 100
 adapter 0 4000.0000.cccc
 adapter 1 4000.0000.dddd
!
interface TokenRing0/0
 no ip address
 ring-speed 16
```

```
 source-bridge 201 1 200
 source-bridge spanning
!
interface Virtual-TokenRing0
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 102 1 100
 source-bridge spanning
!
interface Virtual-TokenRing1
 mac-address 4000.1234.1088
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 202 1 200
 source-bridge spanning
!
interface Virtual-TokenRing2
 mac-address 4000.1234.1089
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 203 1 200
!
snasw cpname NETA hostname
snasw dlus NETA.HOSTCMC1 backup NETA.HOSTCMC2
snasw port CIP Virtual-TokenRing0
snasw port DOWNSTRM Virtual-TokenRing1 conntype no-hpr
snasw port DOWNSTRM Virtual-TokenRing2 conntype no-hpr
snasw link HOSTCMC1 port CIP rmac 4000.0000.cccc
snasw link HOSTCMC2 port CIP rmac 4000.0000.dddd
```

SNASw over HPR/IP Configuration Example

Figure 227 illustrates a basic SNASw link over HPR/IP on the upstream connections to the host. The downstream devices connect through Token Ring 0/0.

Figure 227 SNASw over HPR/IP



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

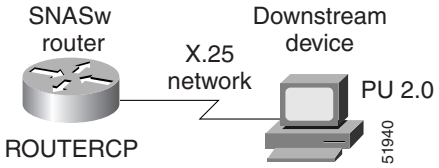
The configuration for SNASw over HPR/IP is as follows:

```
interface Ethernet1/0
 ip address 172.18.49.28 255.255.255.0
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
!
interface TokenRing0/0
 mac-address 4000.1234.1088
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ring-speed 16
!
snasw cpname NETA.ROUTERCP
snasw dlus NETA.HOSTCMC1 backup NETA.CMCHOST2
snasw port HPRIP hpr-ip Ethernet1/0
snasw port TOK0 TokenRing0/0
snasw link HOSTCMC1 port HPRIP ip-dest 172.18.51.1
snasw link HOSTCMC2 port HPRIP ip-dest 172.18.51.2
```

SNASw Using Local Switching with QLLC Configuration Example

Figure 228 illustrates a basic SNASw link using local switching with QLLC.

Figure 228 SNASw using Local Switching with QLLC



Note This figure and example show only the configuration related to the downstream QLLC device. Upstream connectivity is not shown in this configuration.

In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

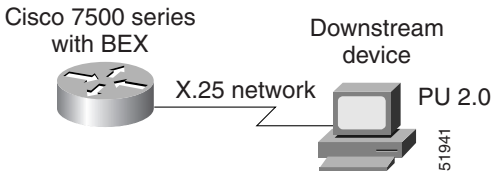
The configuration for SNASw link using Local Switching with QLLC is as follows:

```
!
source-bridge ring-group 70
dlsw local-peer
!
interface Serial4/0
 no ip address
 no ip directed-broadcast
 encapsulation x25
 no ip mroute-cache
 no keepalive
 qllc accept-all-calls
 clockrate 19200
 qllc dlsw vmacaddr 4000.1111.1111 partner 4000.2222.2222
!
snasw cpname NETA.ROUTERCP
snasw port VDLCP vdlc 70 mac 4000.2222.2222 conntype nohpr
```

SNASw Using Local Switching with SDLC Configuration Example

Figure 229 illustrates a basic SNASw link using local switching with SDLC.

Figure 229 SNASw using Local Switching with SDLC



Note This figure and example show only the configuration related to the downstream SDLC device. Upstream connectivity is not shown in this configuration.

In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

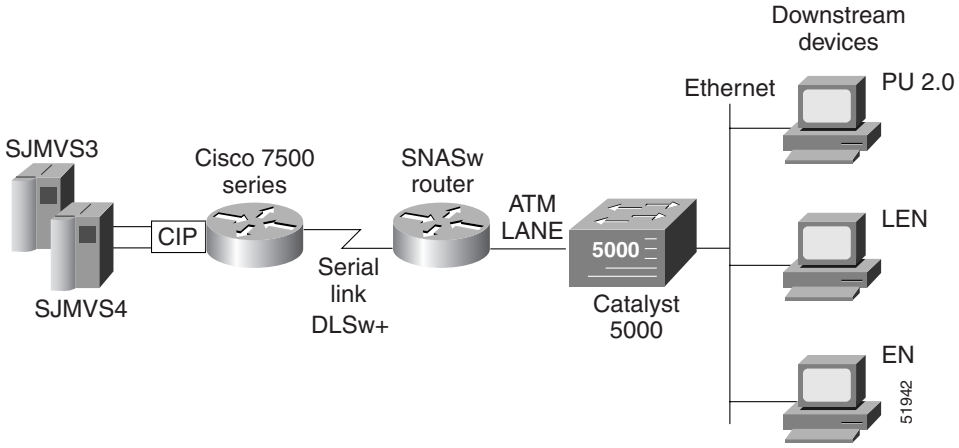
The configuration for SNASw link using local switching with SDLC is as follows:

```
!
source-bridge ring-group 1689
dlsw local-peer
!
interface Serial1
 no ip address
 no ip directed-broadcast
 encapsulation sdlc
 no ip route-cache
 no ip mroute-cache
 no keepalive
 clockrate 9600
 sdlc role primary
 sdlc vmac 4000.3174.0000
 sdlc address C2
 sdlc sdlc-largest-frame C2 521
 sdlc xid C2 05DABBBA
 sdlc partner 4000.4500.00f0 C2
 sdlc dlsw C2
!
snasw cpname NETA.ROUTERCO
snasw port SDLC vdlc 1689 mac 4000.4500.00f0
```

SNASw with Ethernet LAN Emulation over ATM Configuration Example

In Figure 230, downstream devices connect in SNASw over Asynchronous Transfer Mode (ATM) Ethernet LANE. Upstream connectivity is achieved using DLSw+ for connections to the host systems. Downstream devices connect to the standby MAC address on the ATM sub-interface.

Figure 230 SNASw with Ethernet LANE over ATM



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

The configuration for SNASw with Ethernet LANE over ATM is as follows:

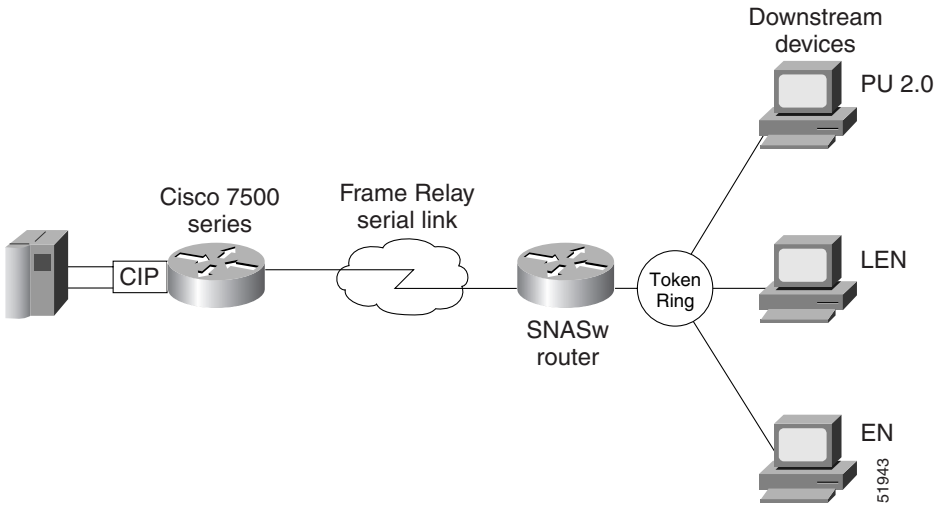
```
!
source-bridge ring-group 111
dlsw local-peer peer-id 10.56.56.1 keepalive 10 promiscuous
dlsw remote-peer 0 tcp 10.56.56.2
!
interface ATM2/0
 mtu 1500
 no ip address
 no ip directed-broadcast
 atm clock INTERNAL
 atm pvc 1 0 5 qsaal
 atm pvc 2 0 16 ilmi
 atm pvc 60 1 36 aal5nlpid
 no atm ilmi-keepalive
!
interface ATM2/0.1 multipoint
 no ip directed-broadcast
 lane client ethernet RED
 no cdp enable
!


interface ATM2/0.2 multipoint
 ip address 10.10.50.60 255.255.255.0
 no ip redirects
 no ip directed-broadcast
 lane client ethernet BLUE
 no cdp enable
```

```
 standby 1 priority 200 preempt
 standby 1 authentication xxxx
 standby 1 mac-address 000b.e291.0000
 standby 1 ip 10.10.50.70
!
interface Serial3/1
 ip address 10.56.56.1 255.255.255.0
 no ip directed-broadcast
 encapsulation ppp
 no keepalive
 no fair-queue
 clockrate 56000
!
snasw cpname NETA.ROUTERCP
snasw dlus NETA.SJMVS3 backup NETA.HOSTCMC2
snasw port ATM202 ATM2/0.2 conntype nohpr
snasw port DLSWP vdlc 111 mac 4000.0189.0016 conntype nohpr
snasw link HOSTCMC1 port DLSWP rmac 4000.aaaa.cccc
snasw link HOSTCMC2 port DLSWP rmac 4000.aaaa.dddd
!
```

SNASw with SRB Frame Relay (Frame Relay BAN Support) Configuration Example

Figure 231 illustrates how to combine SNASw and SRB over Frame Relay functionality to provide native RFC 1490 connectivity over Frame Relay BAN. The host is configured to respond to 4000.aaaa.cccc through the Frame Relay connection over Serial1. Downstream would be configured to connect into Virtual TokenRing0.

Figure 231 SNASw with SRB Frame Relay (Frame Relay BAN Support)



Note In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

The configuration for SNASw with SRB Frame Relay (Frame Relay BAN Support) is as follows:

```
source-bridge ring-group 100
source-bridge ring-group 200
!
interface TokenRing0
no ip address
no ip directed-broadcast
ring-speed 16
source-bridge 202 1 200
!
interface Virtual-TokenRing0
mac-address 4000.1234.1001
no ip address
no ip directed-broadcast
ring-speed 16
source-bridge 201 1 200
!
interface Serial1
 encapsulation frame-relay
!
interface serial 1.1 point-to-point
 frame-relay interface-dlci 30 ietf
 source-bridge 101 1 100
!
interface Virtual-TokenRing1
 mac-address 4000.1111.2222
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 102 1 100
 source-bridge spanning
!
snasw cpname NETA.ROUTERCP
snasw port frame virtual tokenring 1 conntype nohpr
snasw link HOSTFRAM port FRAME rmac 4000.aaaa.cccc
```

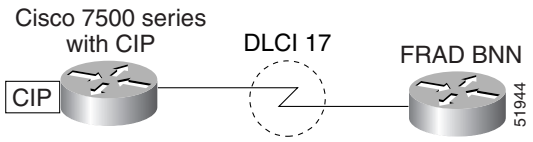
On the CIP router, configure the following:

```
source-bridge ring-group 300
interface serial 1/0
 encapsulation frame-relay
!
interface serial 1/0.1 point-to-point
 frame-relay interface 30 ieft
 source-bridge 101 1 300
!
interface channel 2/1
 no ip-address
 no keep alive
 csna E040 70
!


interface serial channel 2/2
 no ip-address
 no keep alive
lan tokenring 0
 source-bridge 301 1 300
 adapter 0 4000.aaaa.cccc
```

SNASw with FRAS Host (Downstream Frame Relay BNN Support) Configuration Example

Figure 232 illustrates how to connect a downstream Frame Relay BNN device (Frame Relay Access Device) over native RFC 1490 in SNASw.

Figure 232 SNASw with FRAS Host (Downstream Frame Relay BNN Support)



Note This figure and example show only the configuration related to downstream Frame Relay BNN Support. Upstream connectivity is not shown in this configuration segment.

In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

The configuration SNASw with FRAS Host (Downstream Frame Relay BNN Support) is as follows:

```
source-bridge ring-group 200
interface serial 1/2
 no ip-address
 encapsulation frame-relay letf
 frame-relay map llc2 17
!
interface virtual-tokenring 0
 mac-address 4000.1234.1001
 ring-speed 16
 source-bridge 201 1 200
!
interface virtual-tokenring 1
 ring-speed 16
 source-bridge 202 1 200
 fras-host bnn serial 1/2 fr-lsap 04 umac 4000.1234.2002 hmac 4000.1234.1001
```

SNASw vs APPN Connecting VTAM to the CIP Using CMPC Configuration Example

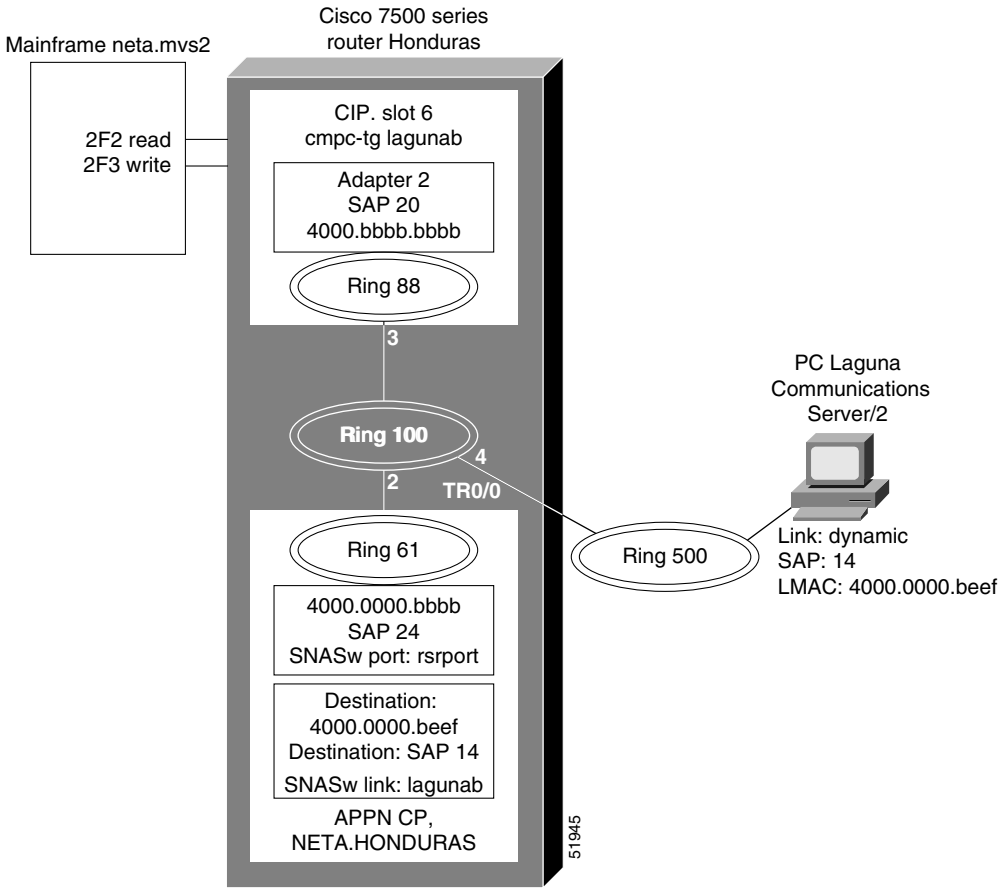
The following section compares the configuration of SNASw vs APPN connecting VTAM to the CIP using CMPC.

Note SNASw supersedes all functionality previously available in the APPN feature in the Cisco IOS software. SNASw configuration will not accept the previous APPN configuration commands and APPN is no longer supported. Previous APPN users should use this chapter to configure APPN functionality using the new SNASw commands.

In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

Figure 233 illustrates the VTAM connecting to SNASw on the CIP using CMPC.

Figure 233 Topology for VTAM-to-SNASw Connection on the CIP



Configuration for TRL Node LAGTRLB

```
LAGTRB   VBUILD TYPE=TRL
```

```
LAGTRLB  TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
               READ=(2F2),                                        X
               WRITE=(2F3)
```

Local SNA Major Node LAGLNB

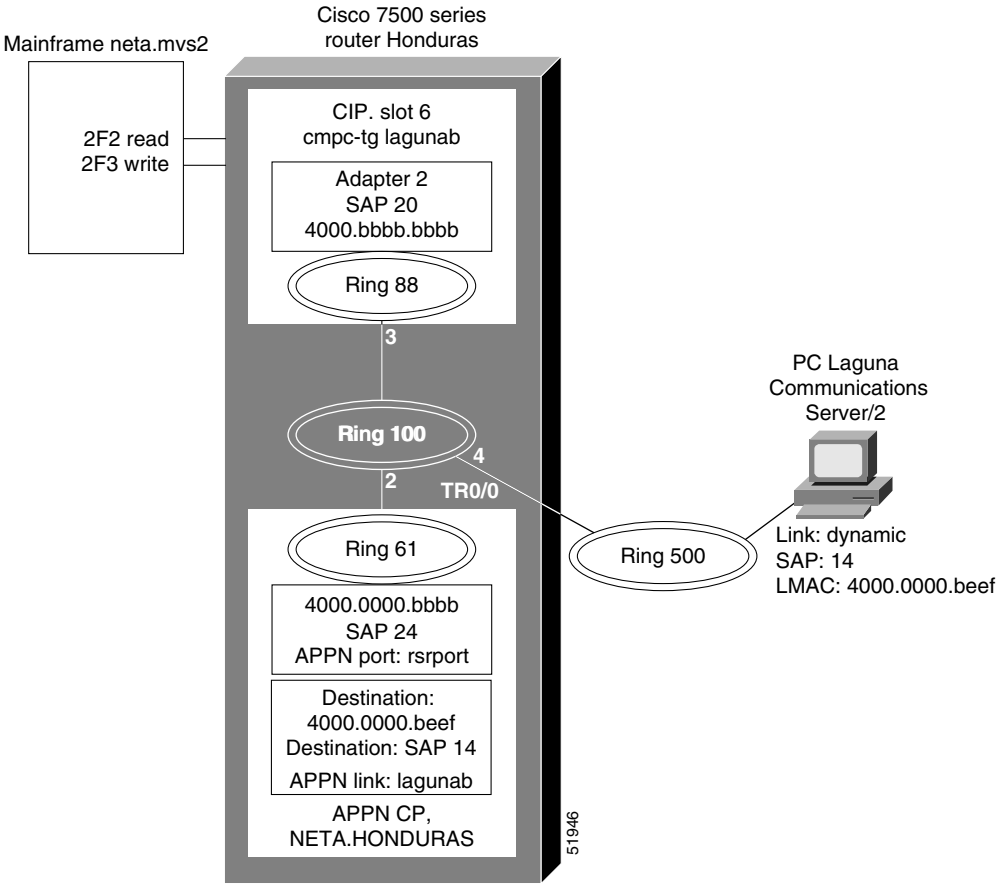
```
LAGNNB   VBUILD TYPE=LOCAL
LAGPUB   PU    TRLE=LAGTRLB,                                      X
               ISTATUS=ACTIVE,                                    X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

Honduras Router

```
source-bridge ring-group 100
!
interface Channel6/1
 no ip address
 no keepalive
 cmpc C020 F2 LAGUNAB READ
 cmpc C020 F3 LAGUNAB WRITE
!
interface Channel6/2
 no ip address
 no keepalive
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 2 4000.bbbb.bbbb
 lan TokenRing 2
 tg LAGUNAB  llc token-adapter 2  20 rmac 4000.0000.bbbb rsap 24
!
!
interface Virtual-TokenRing0
 mac-address 4000.0000.bbbb
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 61 2 100
!
snasw cpname NETA.HONDURAS
snasw port VTOK Virtual-TokenRing0
snasw link MVS2D port VTOK rmac 4000.bbbb.bbbb
```

By comparison, Figure 234 illustrates the VTAM connecting to the APPN NN on the CIP using CMPC.

Figure 234 Topology for VTAM-to-APPN NN Configuration on the CIP



Configuration for TRL Node LAGTRLB

```
LAGTRB   VBUILD TYPE=TRL
LAGTRLB  TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
               READ=(2F2),                                         X
               WRITE=(2F3)
```

Local SNA Major Node LAGLNB

```
LAGNNB   VBUILD TYPE=LOCAL
LAGPUB   PU    TRLE=LAGTRLB,                                       X
               ISTATUS=ACTIVE,                                     X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

Honduras Router

```
interface Channel6/1
 no ip address
 no keepalive
 cmpc C020 F2 LAGUNAB READ
 cmpc C020 F3 LAGUNAB WRITE
!
interface Channel6/2
 no ip address
```

```
 no keepalive
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 2 4000.bbbb.bbbb
 lan TokenRing 2
 tg LAGUNAB  llc token-adapter 2  20 rmac 4000.0000.bbbb rsap 24
!
appn control-point NETA.HONDURAS
    complete
!
appn port RSRBPORT rsrb
  local-sap 24
  desired-max-send-btu-size 4096
  max-rcv-btu-size 4096
  rsrb-virtual-station 4000.0000.bbbb 61 2 100
  complete
!
appn link-station LAGUNAB
  port RSRBPORT
  lan-dest-address 4000.0000.beef 14
  complete
router eigrp 109
network 172.18.0.0
```

SNASw vs APPN Connecting to VTAM on a Remote Router with DLUR Using CMPC

The following section compares the configurations of SNASw vs APPN while connecting to VTAM on a remote router with DLUR using CMPC.

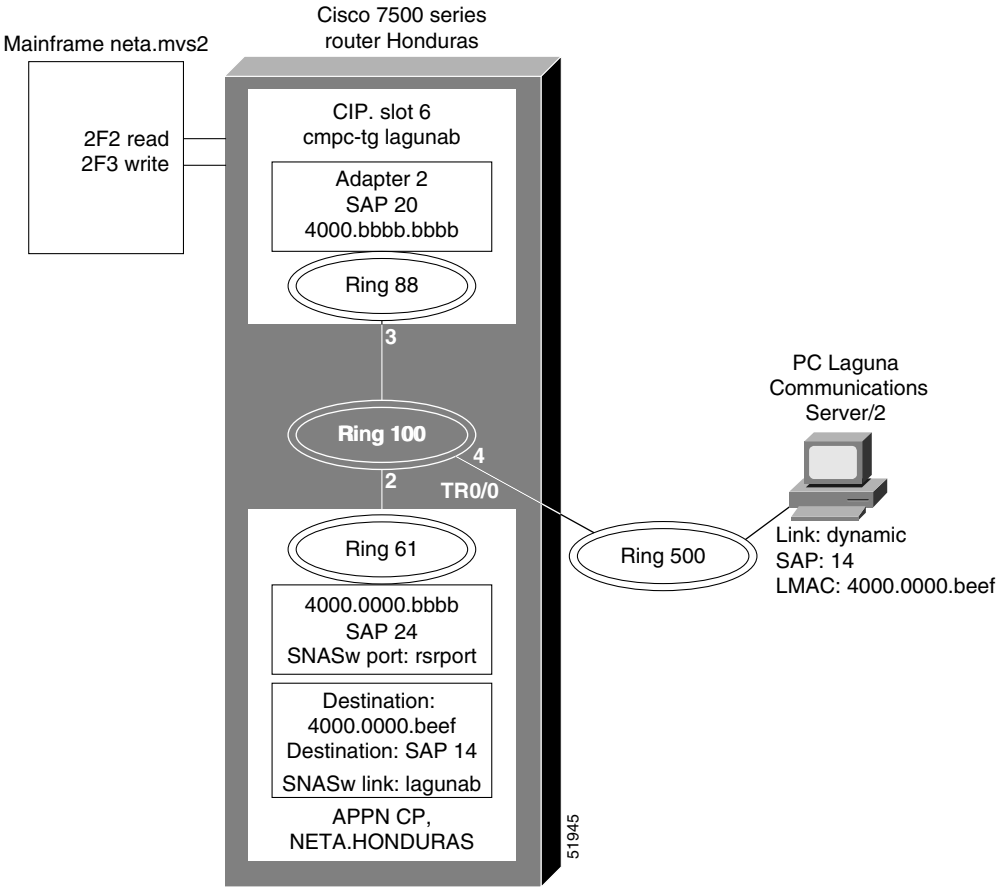
In the example shown in Figure 235 and Figure 236, DLUS is running on the MVS host. DLUR is running on a remote Cisco 4000 router. The connection from MPC to the APPN stack on the Cisco 4000 is via LLC2. There is no NN on the Cisco 7500. The PC is running Communications Server/2.

Note SNASw supersedes all functionality previously available in the APPN feature in the Cisco IOS software. SNASw configuration will not accept the previous APPN configuration commands and APPN is no longer supported. Previous APPN users should use this chapter to configure APPN functionality using the new SNASw commands.

In a typical configuration, downstream links are not configured. Instead, the downstream device configures the connection to the SNASw router, and the SNASw router creates a dynamic link definition for the downstream links.

Figure 235 illustrates a DLUS-to-DLUR configuration using SNASw.

Figure 235 Topology for VTAM-to-SNASw on a Remote Router with DLUR Connection



mvs2trld

```
MVS2TRD  VBUILD TYPE=TRL
MVS2TRLD TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
               READ=(2F7),                                         X
               WRITE=(2F6)
```

mvs2lnd

```
MVS2NND  VBUILD TYPE=LOCAL
MVS2PUD  PU    TRLE=MVS2TRLD,                                      X
               ISTATUS=ACTIVE,                                     X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

Additional Configuration for Router Honduras

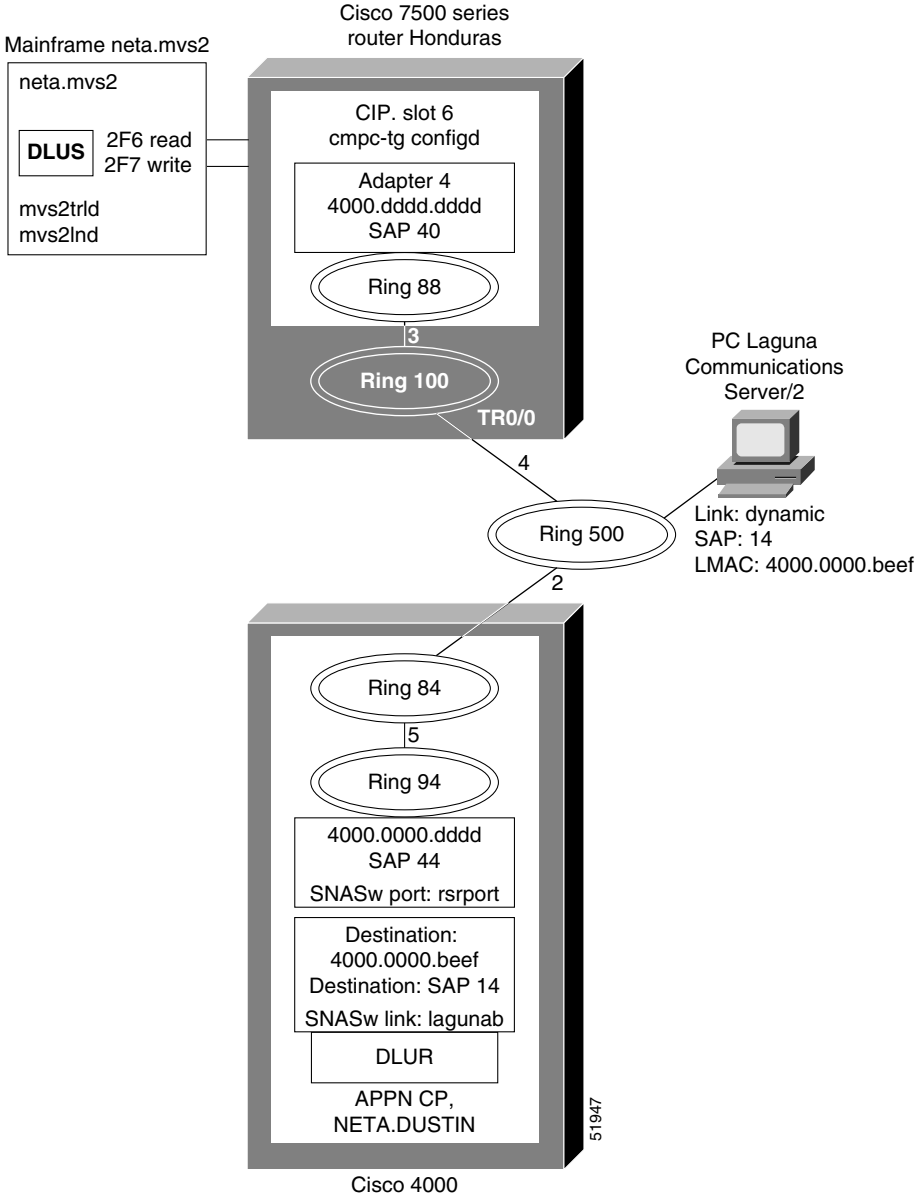
```
interface Channel6/1
 cmpc C020 F6 CONFIGD WRITE
 cmpc C020 F7 CONFIGD READ
!
interface Channel6/2
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 4 4000.dddd.dddd
  tg CONFIGD  llc token-adapter 4  40 rmac 4000.0000.dddd rsap 44
```

Router Dustin

```
source-bridge ring-group 84
interface Ethernet0
 ip address 172.18.3.36 255.255.255.0
 media-type 10BaseT
!
interface TokenRing0
 no ip address
 ring-speed 16
 source-bridge 500 2 84
!
interface Virtual-TokenRing0
 mac-address 4000.0000.dddd
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 94 5 84
!
snasw cpname NETA.DUSTIN
snasw dlus NETA.MVS2
snasw port VTOK Virtual-TokenRing0
snasw link MVS2D port VTOK rmace 4000.dddd.dddd
```

By comparison, Figure 236 illustrates a DLUS-to-DLUR configuration using APPN.

Figure 236 Topology for VTAM-to-APPN NN on a Remote Router with DLUR Connection



mvs2trld

```
MVS2TRD  VBUILD TYPE=TRL
MVS2TRLD TRLE   LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
                READ=(2F7),                                         X
                WRITE=(2F6)
```

mvs2lnd

```
MVS2NND  VBUILD TYPE=LOCAL
MVS2PUD  PU     TRLE=MVS2TRLD,                                      X
                ISTATUS=ACTIVE,                                     X
                XID=YES,CONNTYPE=APPN,CPCP=YES
```

Additional Configuration for Router Honduras

```
interface Channel6/1
 cmpc C020 F6 CONFIGD WRITE
 cmpc C020 F7 CONFIGD READ
!
interface Channel6/2
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 4 4000.dddd.dddd
  tg CONFIGD  llc token-adapter 4  40 rmac 4000.0000.dddd rsap 44
```

Router Dustin

```
source-bridge ring-group 84
interface Ethernet0
 ip address 172.18.3.36 255.255.255.0
 media-type 10BaseT
!
interface TokenRing0
 no ip address
 ring-speed 16
 source-bridge 500 2 84
!
appn control-point NETA.DUSTIN
  dlus NETA.MVS2
  dlur
  complete
!
appn port RSRBPORT rsrb
  local-sap 44
  desired-max-send-btu-size 4096
  max-rcv-btu-size 4096
  rsrb-virtual-station 4000.0000.dddd 94 5 84
  complete
!
appn link-station LAGUNAD
  port RSRBPORT
  lan-dest-address 4000.0000.beef 14
  complete
!
appn link-station MVS2D
  port RSRBPORT
  lan-dest-address 4000.dddd.dddd 40
complete
```

SNASw Dial-out to a DLUR Downstream Configuration Example

SNASw downstream connections are usually initiated by downstream devices. Alternatively, VTAM has a“dial-out” function where the downstream connectivity information is configured at VTAM, and passed down to SNASw to initiate the connection from SNASw to the downstream device. This section describes the SNASw and VTAM changes that are needed for this configuration.

The following example shows how to configure SNASw on SNA7200:

Router SNA7200

```
source-bridge ring-group 84
interface FastEthernet0/0
```

```
 ip address 172.18.3.36 255.255.255.0
!
interface TokenRing0
 no ip address
 ring-speed 16
 source-bridge 500 2 84
!
snasw cpname NETA.SNA7200
snasw dlus NETA.MVSD
snasw port FA01EE hpr-ip FastEthernet0/0
snasw port TOK0 TokenRing4/0 conntype dialoutlen
snasw link MVSDIP port FA01EE ip-dest 172.18.1.41
```

Note "conntype dialoutlen" on the downstream port definition (TOK0) is needed only when LU 6.2 communications is used by the downstream device.

The VTAM switched major node syntax is as follows:

```
PATH DLURNAME=<NETID.CP name of SNASw router>,
     DLCADDR=(1,C,TR), <specifies token ring address format>
     DLCADDR=(2,X,<name of downstream port in SNASw config, expressed in hex EBCDIC
format>),
-OR -
     DLCADDR=(2,C,<name of downstream port in SNASw config expressed in character
format>),
     DLCADDR=(3,X, <sap value, usually 04>),
     DLCADDR=(4,X,<mac address of the downstream device in token ring format>)
```

The following example shows how to configure VTAM for a downstream dial out PU where the downstream device on a Token Ring has the MAC address of 1000.5a6d.32ab:

```
EXPUPATH PATH DLURNAME=NETA.SNA7200,
              DLCADDR=(1,C,TR),
              DLCADDR=(2,C,TOK0),
              DLCADDR=(3,X,04),
              DLCADDR=(4,X,10005A6D32AB)
```

In Cisco IOS Release 12.3(14) and earlier, you must use hexadecimal EBCDIC format for DLCADDR parameter 2, as such:

```
EXPUPATH PATH DLURNAME=NETA.SNA7200,
              DLCADDR=(1,C,TR),
              DLCADDR=(2,X,E3D6D2F0),
              DLCADDR=(3,X,04),
              DLCADDR=(4,X,10005A6D32AB)
```

coincidental.

Configuring Cisco Transaction Connection

Configuring Cisco Transaction Connection

This chapter describes how to configure the Cisco Transaction Connection (CTRC) feature. For a complete description of the CTRC commands mentioned in this chapter, refer to the "Cisco Transaction Connection Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2).To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

This chapter contains the following sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Technology Overview

CTRC provides TCP/IP end-users and servers with fast, reliable, and secure access to IBM DB2 databases and Customer Information Control System (CICS) transaction programs. The CTRC feature of the Cisco router provides a flexible, cost-effective, and scalable solution for enterprise-wide database access and transaction processing. CTRC allows Windows or UNIX client applications to call CICS

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

transactions without changes to the client or host software. Any client running a Distributed Relational Database Architecture (DRDA) requestor, which is included in most Open Database Connectivity (ODBC) applications, can use CTRC to access data in DB2 databases.

With CTRC, you can continue using current CICS client/server applications on a more robust, higher-performing platform than the general-purpose operating system gateways. CTRC provides protocol independence between client workstations and the host, enabling the applications to communicate directly with CICS and DB2 without costly mainframe application upgrades or expensive middleware servers.

The CTRC software feature provides:

- Access to DB2 databases from TCP/IP clients
- Access to CICS applications from TCP/IP clients
- A keepalive timer to maintain the TCP/IP connection
- Integration with the Cisco IOS software to provide intelligent network services for application connectivity, workload management, and fault tolerance

CTRC is a standards-based solution that can be managed either from the host, using mainframe management software, or from a Simple Network Management Protocol (SNMP) workstation. The following MIBs allow monitoring the CTRC router from the management platform of choice:

- CISCO-DATABASE-CONNECTION-MIB.my - 93
- CISCO-TRANSACTION-CONNECTION-MIB.my - 144

For descriptions of supported MIBs and how to use MIBs, see the Cisco MIB website on Cisco.com.

Using CTRC for CICS Access

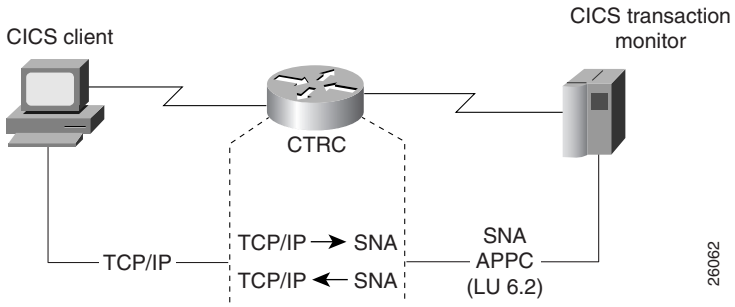
When a router is configured to use CTRC for communications with CICS systems, the router converts Inter-System Communications (ISC) packets over TCP/IP to ISC packets over Advanced Program-to-Program Communications (APPC) LU 6.2, and then routes them to the appropriate CICS region. CTRC converts CICS client messages received via TCP/IP to SNA messages and uses Cisco SNA Switching Services (SNASw) to send them to the host.

When a client connects to a CICS region on an IBM mainframe host, CTRC allocates an APPC conversation over SNA to an IBM server and acts as a gateway between ISC over TCP/IP and ISC over APPC. CTRC allows you to configure specific routes for CICS transactions, giving you control over which transaction is routed to which CICS region.

CTRC supports connectivity to CICS from the IBM Universal Client (also referred to as the Common Client), TXSeries clients, and Microsoft Common Object Module Transaction Interface (COMTI) clients. See the "Configuration CTRC Task List" section on page 4 for details on the hardware and software that CTRC supports.

Figure 237 illustrates how CTRC allows CICS client applications on TCP/IP networks to interact with CICS transaction monitoring systems on IBM hosts.

Figure 237 Cisco Router Configured with the CTRC Feature for CICS Communications



Using CTRC for DB2 Access

In addition to its CICS-related functionality, CTRC includes the feature previously known as Cisco Database Connection (CDBC). CTRC allows Cisco routers to use IBM's DRDA protocol to provide a gateway between client workstations on TCP/IP networks and IBM DB2 databases on SNA networks. CTRC also provides full duplex TCP passthrough to DB2 systems that support direct TCP/IP access.

Clients use a CTRC IP address and port on the router to connect to the IBM host system in either an SNA network or a TCP/IP network.

Figure 238 illustrates how the Cisco router configured with the CTRC feature enables the exchange of database information between an ODBC client application running DRDA in a TCP/IP network and a DB2 system in an SNA network. For an SNA host connection, the CTRC router converts DRDA packets over TCP/IP to DRDA packets over APPC (LU 6.2) and then routes them to DB2 databases. When a client connects to the database on an IBM mainframe host, CTRC allocates an APPC conversation over SNA to an IBM server and acts as a gateway between DRDA over TCP/IP and DRDA over APPC.

Figure 238 Cisco Router Configured with the CTRC Feature for DB2 Communications (SNA Host Network)

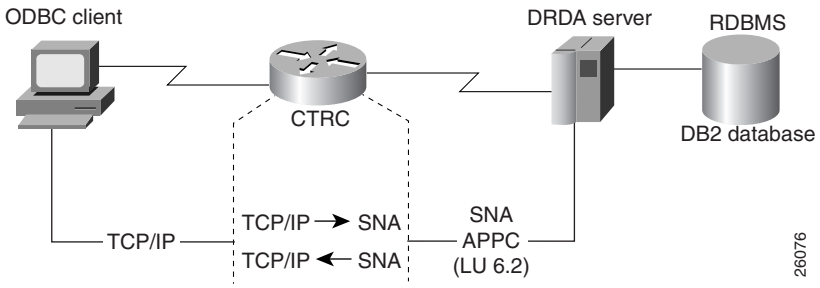
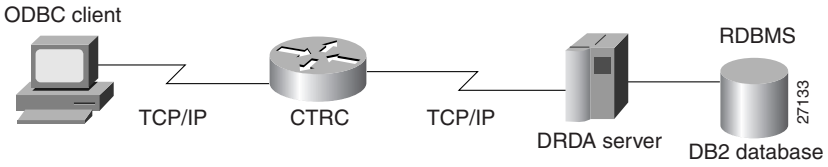


Figure 239 illustrates a configuration where CTRC supports direct TCP/IP access to DB2. For a TCP/IP host connection, CTRC routes the DRDA packets over TCP/IP without protocol changes. To use this TCP/IP passthrough feature of CTRC, the host database version must support direct TCP/IP access and the SNA Switching Services must be available.

Note Licensing of the CTRC router is based on the cpname assigned to the router in the SNA Switching Services configuration. You must install and start SNA Switching Services with at least a minimal configuration to support the TCP/IP connections. Refer to the "Configuring SNA Switching Services" section on page 17 for more information about configuring the CTRC license and the SNA Switching Services that CTRC requires.

Figure 239 Cisco Router Configured with the CTRC Feature for DB2 Communications (TCP/IP Host Network)



Using the CTRC Keepalive Timer

In environments where there is heavy network traffic or limited processing capabilities, TCP/IP connections can time out before transactions are completed. The Keepalive Timer feature enables CTRC servers to send acknowledgment messages to clients at specific intervals to maintain the TCP/IP connection. CTRC servers that support direct TCP/IP connections to a DB2 host also can be configured to send keepalive messages to the host. The Keepalive Timer feature keeps TCP/IP connections active so they do not time out from inactivity.

Configuration CTRC Task List

CTRC can be configured for use with CICS, with DB2, or both. Both CICS and DB2 configurations require Cisco SNA Switching Services.

General Tasks

Setting up CTRC involves the following general tasks:

- Defining the CTRC Router to VTAM, page 6
- Preparing a CICS Host for Remote Access, page 7
- Preparing a DB2 Host for Remote Access, page 11
- Configuring the CTRC Router, page 15
- Verifying the CTRC Configuration, page 18
- Configuring CTRC Clients, page 21

To configure CTRC for use with both CICS and DB2, complete all the configuration tasks. Otherwise, skip the sections that are related only to CICS or DB2, as appropriate for your needs. The "CTRC Configuration Examples" section on page 27 provides example configurations for using CTRC in various network topologies.

The following sections describe the hardware and software required to use CTRC.

Router Requirements

CTRC became available in Cisco IOS Release 12.05(XN). It is available for the following platforms:

- Cisco 7200 series routers
- Cisco 7500 series routers

CTRC consists of a system image and a microcode image, which are virtually bundled as one combined image. Within the Cisco IOS software listings, look for a software feature called Enterprise/SNASw Plus.

If you want to run CTRC on a router with a CIP card, also be sure to download the CIP hardware microcode appropriate for the Cisco IOS software level you are using.

Host Requirements

Mainframe hosts using SNA with the CTRC server must be running VTAM V3.0 or later.

CICS Host Requirements

Using CTRC for CICS access requires CICS Version 4.0 or later. CTRC supports the following CICS servers:

- CICS Transaction Server for OS/390, Version 1 or later
- CICS/400, Version 3.1
- CICS on Open Systems and NT (TXSeries)
- CICS/ESA, Version 3.3*
- CICS/ESA, Version 4.1
- CICS/MVS, Version 2.12.*
- CICS/VSE, Version 2.2*
- CICS/VSE, Version 2.3
- CICS for OS/2, Version 2.01 or later

Note Versions marked with an asterisk (*) have limited server support. These versions support ECI but they *do not* support EPI or the Terminal Emulation function.

DB2 Host Requirements

When CTRC is configured for access to DB2 in an SNA network, client-based ODBC applications can connect to the following IBM DB2 relational databases:

- DB2 for OS/390 (DB2/MVS), Version 2.3 or later
- SQL/DS (DB2 for VM and VSE), Version 3.3 or later
- DB2/400 (OS/400), Version 2.2 or later
- DB2 Universal Database for UNIX, OS/2, and Windows NT, Version 5.1 or later
- DB2 Common Server, Version 2.1 or later

CTRC for DB2 access via direct TCP/IP is supported for the following versions of DB2:

- DB2 for OS/390, Version 5.1 or later (requires OS/390 Version 1.3 or later)
- DB2 for VM and VSE, Version 6.1 or later
- DB2/400 (OS/400), Version 4 Release 2 or later
- DB2 Universal Database for UNIX, OS/2, and Windows NT, Version 5.1 or later

Client Requirements

CTRC supports connectivity to DB2 from any client that supports the Level 3 DRDA. Many of the available workstation-based DRDA requestors are ODBC client applications, such as StarSQL.

CTRC supports connectivity to CICS from the following clients:

- IBM Universal Client, version 2.0 or later, using the Extended Presentation Interface (EPI) or the Extended Call Level Interface (ECI)
- IBM TXSeries for AIX or NT, version 4.2 or later, running as clients
- Microsoft COMTI

Defining the CTRC Router to VTAM

Regardless of whether you want to connect to a CICS or a DB2 host, the CTRC router must be defined to VTAM so that the host recognizes and accepts session initiation requests from it. VTAM handles network communications for MVS for direct VTAM and SNA gateway configurations. For each CTRC router, the VTAM system programmer must create a logmode table entry and major node definitions for the CTRC router link.

The following sections provide information about the logmode table entry and major node definitions required for CTRC. Consult your VTAM documentation for detailed instructions on configuring VTAM. You also may want to take advantage of VTAM's support for dynamic definition of independent LU's, which is described in the VTAM documentation.

Logmode Table Entry

The logmode table entry contains information that governs how conversations take place in VTAM. It defines pacing, RU sizes and class of service (COS) parameters. The mode entry can be placed in any mode table under VTAM—the default mode table or the one used in the APPL statement for the LU definitions. (See the "Defining the CICS Subsystem to VTAM" section on page 7 and the "Defining the DB2 Subsystem to VTAM" section on page 11 for example APPL statements).

The following example shows a logmode table entry for APPC, with a LOGMODE name of IBMRDB. Make a note of the LOGMODE name because you must use the same name for the DLOGMODE value in the major node definitions and also in the SNA configuration. The PSERVIC field identifies the LU traffic protocol—the value shown in the following example is for an independent LU using LU6.2.

```
IBMRDB    MODEENT  LOGMODE=IBMRDB,
                   FMPROF=X'13',
                   TSPROF=X'01',
                   PRIPROT=X'B0',
                   SECPROT=X'B0',
                   COMPROT=X'50A1',
```

```
                    RUSIZES=X'8989',
                    TYPE=0,
                    PSNDPAC=X'03',
                    SRVCPAC=X'03',
                    SSNDPAC=X'02',
                    PSERVIC=X'060200000000000000002F00'
```

Major Node Definitions

The VTAM system programmer creates an XCA major node definition for the connection to the CTRC router. Additionally, a switched major node definition and a Cross Domain Resource definition can be created to represent the LU for the CTRC router.

In the switched major node definition, the DLOGMOD value must match the LOGMODE value in the mode table entry. The name of IBMRDB is specified for both the LOGMODE value in the previous example and in the following switched major node definition example. Make a note of the values for the LU and PU names, and the CPNAME, DLOGMOD, and CONNTYPE parameters because you must specify the same values in the SNA configuration.

```
S02CTRC     VBUILD     TYPE=SWNET
* CTRC  DOWNSTREAM  PU
CTRCPU    PU       ADDR=01,
                   CPNAME=CTRCBOX,
                   ANS=CONT,
                   DISCNT=NO,
                   IRETRY=NO,
                   ISTATUS=ACTIVE,
                   PUTYPE=2,
                   SECNET=NO,
                   MAXDATA=521,
                   MAXOUT=2,
                   MAXPATH=1,
                   USSTAB=USSS,
                   MODETAB=ISTINCLM,
                   DLOGMOD=IBMRDB,
                   CONNTYPE=APPN
*
CTRCCIP  PATH  GRPNM=G02E20A,CALL=IN
*
CTRCBOX  LU       LOCADDR=00,      INDEPENDENT LU
                   DLOGMOD=IBMRDB,
```

Preparing a CICS Host for Remote Access

CTRC connects to CICS using the SNA LU6.2 (APPC) communication protocol. The SNA functions are provided by a separate SNA product on the host, and CICS uses the services of that product. On a mainframe host, the SNA product is VTAM (also known as eNetwork Communications Server). You must configure both the CICS subsystem and VTAM to enable ISC.

Defining the CICS Subsystem to VTAM

The APPL statement defines the CICS subsystem to VTAM to support remote access. If your CICS subsystem is not already supporting remote access, you must create an appropriate APPL statement.

The following example shows an APPL statement that defines CICS to VTAM. Make a note of the APPL statement label, which is CICSB in this example, and the password, if one is specified, because you must specify the same values in the SNA configuration. Note that the DLOGMOD value, IBMRDB in this example, must match the LOGMODE value that is specified in the VTAM mode table entry (see the "Logmode Table Entry" section on page 6).

```
A02CICS  VBUILD   TYPE=APPL
CICSB    APPL     AUTH=(ACQ,SPO,PASS,VPACE),
                  MODETAB=ISTINCLM,
                  DLOGMOD=IBMRDB,
                  HAVAIL=YES,
                  VPACING=9,
                  EAS=10000,
                  PARSESS=YES,
                  APPC=NO,
                  SONSCIP=YES
```

Configuring CICS for ISC

To use CTRC to communicate with CICS, you must configure CICS for APPC connections. If you have configured another product, such as TXSeries for AIX, to connect to CICS, some of these steps might be completed already.

Step 1 Set the ISC parameter in the CICS system initialization table (SIT) to YES. The following example overrides the CICS SIT parameters with the APPL statement label (CICSB in this example), and a value of YES for the ISC parameter.

```
APPLID=(CICSB),
GMTEXT='CICS TS V1.2',
AUXTR=OFF,
EDSALIM=80M,
FCT=NO,
ISC=YES,
MXT=100
```

Step 2 Install the CICS-supplied resource definition group, DFHCLNT. This installation includes definitions of the CICS internal transactions, CCIN and CTIN, and of the programs they use.

Step 3 When a CICS client sends a request, the server controller calls a routine that supports code page translations and data conversions. Regardless of whether translations and conversions are required, you need to create or modify a DFHCNV table to allow the server controller to handle incoming requests. The use of the DFHCNV macro for defining the table is described in the *CICS Family, Communicating from CICS on System/390* document. The following example shows the DFHCNV table entries:

```
PRINT   NOGEN
DFHCNV   TYPE=INITIAL,SRVERCP=037,CLINTCP=437
DFHCNV   TYPE=FINAL
END     DFHCNVBA
```

Note It is not necessary to code the pages used with CICS clients on the CLINTCP and SRVERCP operands of the DFHCNV TYPE=INITIAL macro.

Step 4 Messages relating to client support are written to the CSCC transient data queue, which you must define to CICS. There is a sample definition in the supplied resource definition group, DFHDCTG. The sample defines CSCC as an indirect extra partition destination, pointing to CSSL.

Defining APPC Connections to CTRC

You must install APPC connections to define the CTRC connection to CICS. This section describes the definitions and methods for installing them.

In the CONNECTION definition you specify information about the CTRC router and how it connects to CICS. The following example shows a CONNECTION definition named CTRC. Note that the NETNAME value must be the same as the CTRC router LU name, which is CTRCBOX in this example. Setting the AUTOCONNECT option to YES allows CICS to dynamically activate the router connection. See the "Supporting CICS Security Models" section on page 10 for information about specifying security parameters in the CONNECTION definition.

```
DEFINE
       CONNECTION(CTRC)
       DESCRIPTION(CTRC)
       AUTOCONNECT(YES)
       NETNAME(CTRCBOX)
       ACCESSMETHOD(VTAM)
       PROTOCOL(APPC)
       SINGLESESS(NO)
       ATTACHSEC(IDENTIFY)
       BINDPASSWORD(NO)
       BINDSECURITY(NO)
       USEDFLTUSER(YES)
```

Following is an example SESSIONS definition. Note that the value for the CONNECTION parameter must be the same as the name of the CONNECTION definition, which is CTRC for this example.

```
DEFINE
       SESSIONS(CTRC)
       CONNECTION(CTRC)
       MODENAME(IBMRDB)
       PROTOCOL(APPC)
       MAXIMUM(64,1)
       SENDSIZE(4096)
       RECEIVESIZE(4096)
```

The connections can be single- or parallel-session links. Install APPC connections to CICS either by creating static definitions for the router or using an autoinstall. The installation methods are addressed in the following sections.

Creating Static Definitions for Router Connections

You can use the CICS CEDA transaction DEFINE and INSTALL commands to create static definitions. For more information about defining APPC connections, refer to the *CICS Intercommunication Guide*.

Using Autoinstall for Router Connections

Another method of installing router connections is to use autoinstall. If you use autoinstall you must create suitable CONNECTION and SESSIONS template definitions. For information about autoinstall and defining templates, see the *CICS Resource Definition Guide*. For information about customizing your autoinstall user program to handle APPC connections, see the *CICS Customization Guide*.

Installing Client Virtual Terminals

Virtual terminals are used by the EPI and terminal emulator functions of the CICS client products. Both IBM-supplied autoinstall programs support virtual terminal autoinstall. Refer to the *CICS Customization Guide* for detailed information on autoinstall for virtual terminals.

Supporting CICS Security Models

This section addresses how to configure the the Bind, Link, and User security models that are supported in CICS.

Bind Security

Bind-time security currently cannot be configured on the Cisco router. Therefore, specify BINDSECURITY(NO) in the CONNECTION definitions that define the router to CICS.

Link Security

Link security provides the lowest level of resource security for intercommunication links. It defines the total set of resources that can be accessed across the connection.

To set link security for a CICS client connection, specify a userid for the link for the SECURITYNAME option of the CONNECTION definition. Then define a profile to your External Security Manager for the link userid. Users of the connection will be able to access only those resources that the link userid is authorized to access.

If you do not specify a userid for the SECURITYNAME option, the authority of the link is that of the CICS default user.

User Security

User (attach-time) security defines how individual users of an intercommunication link are to be checked. It also affects the resources that individual users are able to access. Unless you specify LOCAL user security (in which case all potential users share the authority of the link userid), you must define user profiles to your External Security Manager.

Preparing a DB2 Host for Remote Access

CTRC provides a gateway between DRDA client requests over TCP/IP to DB2 in SNA networks. CTRC also provides full duplex TCP passthrough to DB2 systems that support direct TCP/IP access. Perform the steps in this section if you want to use CTRC to provide access to DB2 hosts. Otherwise, skip to the "Configuring the CTRC Router" section on page 15.

Defining the DB2 Subsystem to VTAM

The APPL statement defines the DB2 subsystem to VTAM to support remote access. If your DB2 system is not already supporting remote access, you must create an appropriate APPL statement.

The following is an example of an APPL statement. Make a note of the APPL statement label, which is DSNV510 in the following example, and the password, if one is specified. You need to specify the same values when you configure or update the distributed data facility (DDF) record in the Bootstrap Data Set (BSDS) as described in the next section.

```
DB2APPL    VBUILD    TYPE=APPL
DSNV510    APPL      AUTH=(ACQ),
                     APPC=YES,
                     AUTOSES=1,
                     DMINWNL=10,
                     DMINWNR=10,
                     DSESLIM=20,
                     MODETAB=ISTINCLM,
                     SECACPT=ALREADYV,
                     SRBEXIT=YES,
                     VERIFY=NONE,
                     VPACING=2
```

Configuring DB2 for Remote Access

To use CTRC as a gateway between TCP/IP clients and the DB2 host, you need to configure and start DDF and define the CTRC router in the DB2 communications database table.

Configuring DDF

DB2 reads the BSDS during start up to obtain the system installation parameters. The DDF record in the BSDS contains information used by DB2 to connect to VTAM. If the DB2 system supports direct TCP/IP access, the DDF record specifies which port to use for TCP/IP communications.

If you are installing DB2, use the DDF installation panel DSNTIPR to provide the following parameters. If DB2 is already installed, use the change log inventory utility DSNJU003 to update this information in BSDS.

- DDF location name
- DDF LUNAME
- Password used when connecting DB2 to VTAM, if a password is required
- IP port to use for TCP/IP access

The following example updates the BSDS with a location name of DB2510, LU name of DSNV510 for SNA access, a password of STARPASS, and a port of 446 for TCP/IP communications. The RESPORT and PORT parameters are required only for TCP/IP access and can be omitted if using only SNA.

```
//*
//DSNTLOG  EXEC  PGM=DSNJU003,COND=(4,LT)
//STEPLIB   DD   DISP=SHR,DSN=DSN510.SDSNLOAD
//SYSUT1    DD   DISP=OLD,DSN=DSN5CAT.BSDS01
//SYSUT2    DD   DISP=OLD,DSN=DSN5CAT.BSDS02
//SYSPRINT  DD   SYSOUT=*
//SYSUDUMP  DD   SYSOUT=*
//SYSIN     DD   *
            DDF  LOCATION=DB2510,LUNAME=DSNV510,
                 PASSWORD=STARPASS,RESPORT=5020,PORT=446
//*
```

LOCATION is used as the Remote Database (RDB) name. If your system does not require a password to connect DB2 to VTAM, replace the PASSWORD parameter with NOPASSWD. Note the DDF LUNAME because you must specify the same value in the SNA configuration. Also make a note of the LOCATION name because you must specify the same value as the Database Server Name during data source configuration on the desktop (described in the "Setting Up DB2 DRDA Client Connections" section on page 21).

Note You also can determine the DDF location name from the syslog. The DB2 message "DSNL004I (starting DDF)" contains the location name.

For complete information about configuring DDF, consult IBM's DB2/MVS installation documentation.

Starting DDF

Use the following command, which requires authority of SYSOPR or higher, to start DDF:

```
-START DDF
```

When DDF starts successfully, the following messages are displayed:

```
DSNL003I - DDF IS STARTING
DSNL004I - DDF START COMPLETE LOCATION locname LU netname.luname
```

If DDF has not been properly installed, the START DDF command fails and displays the following message:

```
DSN9032I - REQUESTED FUNCTION IS NOT AVAILABLE
```

If DDF has already been started, the START DDF command fails and displays the following message:

```
DSNL001I - DDF IS ALREADY STARTED
```

Defining CTRC in the DB2 Communications Database

The DB2 host maintains a database table that defines the network attributes of remote systems. To enable communication between a CTRC client and the DB2 host, there must be an entry in this table. On DB2 for OS/390 or later versions, the name of this table is SYSIBM.LUNAMES. For DB2 on MVS v4.1, the name of this table is SYSIBM.SYSLUNAMES. Table 9 describes the table entry parameters and indicates which are applicable to one or both versions of the table.

Table 9 DB2 Communications Database Table Entry

Parameter	SYSLUNAMES	LUNAMES	Description
LUNAME	Yes	Yes	LUNAME of the remote system. An empty string means that any LU is valid for this row.
SYSMODENAME	Yes	Yes	VTAM login mode name used for DB2 for MVS/ESA intersystem conversations. A blank frame indicates that IBMDB2LM should be used. Use the mode name specified in the logmode table.
ENCRYPTPSWDS	Yes	Yes	Indicates whether passwords exchanged with this partner are encrypted. Use the default value of NO for passing passwords between a client and DB2 host using CTRC.
MODESELECT	Yes	Yes	If 'Y,' the SYSMODESELECT table is used to obtain the mode name for each outbound distributed database request. If not 'Y,' the mode name IBMDB2LM is used for system-directed access requests, and the mode name IBMRDB is used for DRDA requests.
USERNAMES	Yes	Yes	Indicates the level of come-from checking and user ID translation required. It also specifies the security parameters this DB2 for MVS/ESA subsystem uses when requesting data from the remote partner (outbound security requirements). 'I' indicates an "inbound" ID is subject to translation. 'O' indicates an "outbound" ID, sent to the corresponding LUNAME, is subject to translation. 'B' indicates that both inbound and outbound IDs are subject to translation. A blank indicates no translation for inbound or outbound IDs.
USERSECURITY	Yes	—	Network security acceptance options required of the remote system when the DB2 for MVS/ESA system acts as a server for the remote system (inbound security requirements).
SECURITY_IN	—	Yes	Defines the security options that are accepted by this host when an SNA client connects. 'V' for "verify" indicates that the incoming connection request must include a password. 'A' for "already verified" indicates the request does not require a password, although the password is checked if it is sent.
SECURITY_OUT	—	Yes	Defines the security option that is used when local DB2 SQL applications connect to any remote server associated with this LUNAME. 'A' for "already verified" indicates that outbound connection requests contain an authorization id and no password. 'P' for "password" indicates that outbound connection requests contain an authorization id and password. 'R' for "RACF PassTicket" indicates that outbound connection requests contain a userid and RACF PassTicket.

The following command inserts a row into the SYSIBM.SYSLUNAMES table that any LU can use because the value of the LUNAME column is an empty string:

```
INSERT INTO SYSIBM.SYSLUNAMES (LUNAME, SYSMODENAME, USERSECURITY, ENCRYPTPSWDS,
MODESELECT, USERNAMES) VALUES (' ',' ', 'C', 'N', 'N', ' ');
```

The following command inserts a row into the SYSIBM.LUNAMES table that any LU can use:

```
INSERT INTO SYSIBM.LUNAMES (LUNAME, SECURITY_IN, ENCRYPTPSWDS, USERNAMES) VALUES (' ',
'V', 'N', ' ');
```

Configuring Password Expiration Management

Users of DRDA-based applications, such as StarSQL, can change their host password using CTRC's Password Expiration Management (PEM) feature. This feature is supported by CTRC using IP passthrough and APPC. PEM support for IP passthrough is provided by DB2 for OS390 V5 or later. PEM support when using APPC is provided by either APPC/MVS or CICS.

PEM Support for IP Passthrough

There is no CTRC configuration required for PEM support as it is native in DRDA over TCP/IP. However, the DB2 host must be enabled to support PEM. To enable PEM support on DB2 for OS390 V5 or later, you must configure and use extended security using either:

- The DSNTIPR (DDF) panel on the DB2 installation dialog
- A customized configuration job DSNTIJUZ, with the option EXTSEC=YES specified

Refer to the *DB2 Installation Guide* for details on setting up and using extended security.

Note If you are using DB2 for OS390 V5, install the maintenance fix PTF UQ21052. The IBM APAR PQ15977 describes the problems fixed by this PTF. This maintenance fix is not required for later releases.

PEM Support for APPC

The CTRC PEM support over APPC is implemented using SNA architecture TPs. Therefore, CTRC requires that a surrogate subsystem such as APPC/MVS or CICS be used to change passwords. Both APPC/MVS and CICS support the SNA architecture TPs.

To allow PEM support for DB2 connections, use the **dbconn pem** command to turn on PEM support as needed for the CTRC routers handling the connections. In the **dbconn pem** command statement, specify the LU name of the APPC/MVS base configuration. APPC/MVS configuration statements are in SYS1.PARMLIB(APPCPMxx). Consult your MVS systems programmer to obtain the name of the target LU that will be used by CTRC. The PEM support does not require any explicit definitions of the SNA architecture TPs. The following example shows a LUADD statement, such as found in SYS1.PARMLIB.

```
LUADD ACBNAME(MVSLU01) BASE TPDATA(SYS1.APPCTP)
```

The following is an example VTAM APPL definition for the APPC/MVS LU:

```
MVSLU01     APPL       ACBNAME=MVSLU01,     ACBNAME FOR APPC
                       APPC=YES,
                       AUTOSES=0,
                       DDRAINL=NALLOW,
                       DLOGMOD=IBMRDB,
```

```
DMINWNL=5,
DMINWNR=5,
DRESPL=NALLOW,
DSESLIM=10,
LMDENT=19,
PARSESS=YES,
SECACPT=CONV,
SRBEXIT=YES,
VPACING=1
```

Another alternative for providing PEM support is through the CICS support for SNA architecture TPs, which is provided in resource group DFHISC. To use this method, define the connection to CTRC as described in the "Defining APPC Connections to CTRC" section on page 9, and use the CICS APPLID as the rlu value in the **dbconn pem** command.

Configuring the CTRC Router

After you define the CTRC router to VTAM and prepare the CICS and DB2 hosts for remote access, you must configure the router.

Configuring CTRC for CICS Communications

To configure CTRC to communicate with CICS, you must define a destination and specify a particular server process. You also can define specific routes to be used for particular transaction programs.

Configuring a CTRC Destination for CICS

To configure CTRC to communicate with CICS, you must configure a CTRC destination. A CTRC destination is typically a single CICS system defined in terms of its remote LU name and APPC mode. To configure a destination, use the following global configuration command:

Command	Purpose
`Router(config)# txconn destination` *destination-name* `rlu` *rlu-name* `mode` mode-name	Specifies a CICS system with which CTRC will communicate.

If you want to assign more than one CICS system or region to a single CTRC destination name, such as to help balance the workload, repeat the **txconn destination** command with the same destination name and different remote LU and mode values. If a CTRC destination is configured in this way, the CTRC server sends traffic to the destination's defined CICS regions on a rotating basis. A Cisco router can be configured to communicate with multiple CTRC destinations, whether each of those destinations is defined as an individual pair of remote LU and mode values or as a set of such values.

Configuring a CTRC Server for CICS

After you have configured a CICS destination, configure a CTRC server process to handle communications with that CICS system. Additional CTRC servers can be configured on the same router for communications with other CICS destinations. To configure a CTRC server process to communicate with CICS, use the following global configuration command:

Command	Purpose
Router(config)# **txconn server** *server-name* **destination** *destination-name* [**access** {**cics** | **comti**}] [**client-timeout** *minutes*] [**ccsid** *number*] [**host-timeout** *minutes*] [**ipaddress** *ip-address*] [**keepalive attempts** *number*] [**keepalive interval** *seconds*] [**port** *port-number*] [**target** {**cics** | **ims-tm**}] [**window-size** *bytes*][**fold** {**on** | **off**}]	Configures a CTRC server process for communicating with CICS. If you do not supply a port number, CTRC uses the default value of 1435.

When a client attempts to connect to a CTRC server for CICS, the server's port and IP address determine whether that connection is accepted. By default, the CTRC server port for CICS client communications is 1435. You can create multiple CTRC server processes for both CICS and DB2 on one router.

Configuring a CTRC Route for CICS

After you have configured one or more destinations and server processes for communicating with CICS, you have the option of explicitly configuring CTRC routes that will direct traffic to the appropriate destination based on a transaction ID. If you do not explicitly configure CTRC routes, the CTRC server routes traffic to its own defined default destination. To configure a CTRC route, use the following global configuration command:

Command	Purpose
Router(config)# **txconn route** [**server** *server-name*] **tranid** *transaction-id* **destination** *destination-name*	Configures a particular route for traffic with the specified transaction ID.

Configuring CTRC for DB2 Communications

To configure a CTRC server process for APPC communications with DB2, use the **dbconn server** command in global configuration mode. To configure a CTRC server to communicate with an IP-enabled DB2 database, use the **dbconn tcpserver** global configuration command.

Command	Purpose
Router(config)# **dbconn server** *server-name* [**idle-timeout** *minutes*] [**ipaddress** *ip-address*] [**keepalive attempts** *number*] [**keepalive interval** *seconds*] [**mode** *mode*] [**port** *port-number*] [**rdbname** *rdbname*] [**rlu** *remote-lu*] [**tpname** *tp-name*] [**window-size** *bytes*][**wlm** {**off** | **on**}]	Configures a CTRC server for APPC communications with DB2.
Router(config)# **dbconn tcpserver** *server-name* **remote-hostname** *remote-hostname*|**remote-ip** *remote-ipaddress* [**idle-timeout** *minutes*] [**ip** *ip-address*] [**keepalive attempts** *number*] [**keepalive interval** *seconds*] [**port** *port-number*] [**rdbname** *rdbname*] [**remote-keepalive attempts** *number*] [**remote-keepalive interval** *seconds*] [**remote-port** *remote-port*] [**window-size** *bytes*][**wlm** {**off** | **on**}	Configures a CTRC server to communicate with IP-enabled DB2 databases.

When a client attempts to connect to a CTRC server for DB2, the server's port, IP address, and RDB name determine whether that connection is accepted. By default, the CTRC server port for client requests for DB2 communications is 446. You can create multiple CTRC server processes for both CICS and DB2 on one router.

Configuring SNA Switching Services

CTRC uses the SNA Switching Services (SNASw) of the Cisco router. Even if you do not need to convert client messages received over TCP/IP to SNA messages (such as in a TCP/IP passthrough topology), SNASw must be present, and you must specify a CPNAME for the CTRC router. The following command illustrates the minimal SNASw configuration required to enable the CTRC license:

```
snasw cpname netid.cpname
```

To configure basic SNASw, complete the following steps beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# snasw cpname {netid.cpname | netid [hostname | ip-address interface-name]}`	Defines an SNASw control point name. For the *netid.name* variable, specify the fully qualified CP name for the router, which consists of both network ID and cpname.
Step 2	`Router(config)# snasw port portname [hpr-ip | vdlc ring-group mac mac-address] interfacename [conntype nohpr | len | dyncplen] [nns-required] [hpr-sap hpr-sap-value] [max-links link-limit-value] [maxbtu max-btu-size] [sap sap-value] [vnname virtual-node-name] [nns][nostart]`	Associates an SNASw port with an interface.
Step 3	`Router(config)# snasw link linkname port portname rmac mac-address | ip-dest ip-address [rsap sap-value] [nns] [tgp [high | low | medium | secure]] [nostart]`	Configures upstream links.

Note For a LEN-level connection between SNASw and the host, you also need to configure the **snasw location** configuration command for the specific resource names to be contacted on the host. Do not define locations if APPN connectivity is being used between SNASw and the host. See the "Cisco IOS Software Configuration" section on page 32 for an example of the SNASw configuration statements.

For additional information about configuring SNASw, consult the SNA Switching Services chapter of this document.

Configuring the CTRC License

An unlicensed installation of CTRC allows up to two DB2 connections, two CICS conversations, or one DB2 connection and one CICS conversation for evaluation purposes. To use more than two connections or conversations, you must configure the CTRC license.

The CTRC license key is locked to one node and is based on the SNASw control point name (cpname) for the router. Use the **show config | include cpname** command to determine the cpname for the router you want to license. Then contact your Cisco representative and request a CTRC license key. You will receive a license key along with information about the number of connections you are licensing and, if the license has a time limit, the expiration date.

For communications with DB2, CTRC checks the number of connections in use against the licensed number of connections. For communications with CICS, CTRC checks the number of concurrent and queued conversations. One license key is used for both CICS and DB2 communications, so you can use

either of the following global configuration commands to configure the CTRC license. If your license is not for an unlimited number of connections and period of time you must specify the number of connections and expiration date.

Command	Purpose
`Router(config)# dbconn license` *license-key* [**connections** *licensed-connections*] [**expiration-date** *yyyymmdd*]	Configures a CTRC license.
`Router(config)# txconn license` *license-key* [**connections** *licensed-connections*][**expiration-date** *yyyymmdd*]	Configures a CTRC license.

Verifying the CTRC Configuration

After preparing the host systems and configuring the CTRC router, perform the following steps to ensure CTRC can communicate with the host systems:

Step 1 To verify that you have SNA connectivity between the router and each host system, use the **ping sna** command, specifying the mode and the fully-qualified remote LU name appropriate for your environment in place of IBMRDB and STARW.BUDDY in the following example.

```
ping sna -m IBMRDB STARW.BUDDY
```

Step 2 If you configured CTRC for communications with CICS, perform the following steps to verify the router is properly configured. Skip to Step 3 if you are using CTRC only for DB2 communications.

a. Enter the **show txconn destination** command in EXEC or privileged EXEC mode. Make sure that all CICS destinations you configured are listed with the RLU and mode values you specified.

```
Router# show txconn destination
Name               Remote LU          Mode        Hits
------------------ ------------------ ----------- --------
CICSB              CICSB              IBMRDB       0
GEN                CICSB              IBMRDB       0
                   CICSC              IBMRDB       0
GUAVA              GUAVA              IBMRDB       0
CICSC              CICSC              IBMRDB       0
```

b. For each CICS destination shown in the previous step, enter the **txconn ping** command to verify that the router can communicate with that destination.

```
Router# txconn ping CICSB
Trying CICSB CICSB:IBMRDB
Destination CICSB successfully contacted!
Elapsed time was 00:00:00.600
```

c. Enter the **show txconn server** command. Make sure that all CTRC servers you defined for communications with CICS are listed with the configuration values you specified.

```
Router# show txconn server
Server     Port  IP Address   Dest      State     NumConn
---------- ----- ------------ --------- --------- -------
CICSB      1435  0.0.0.0      CICSB     enabled   0
CICSB&C    1436  0.0.0.0      GEN       enabled   0
CICSC      1434  0.0.0.0      CICSC     enabled   0
GUAVA      1437  0.0.0.0      GUAVA     enabled   0
```

Use the **show txconn server** *server-name* form of the command to display detailed information for an individual server.

```
Router# show txconn server CICSB
                server: CICSB
           destination: CICSB
          server state: enabled (accepting connections)
            ip address: 0.0.0.0
                  port: 1435
        client timeout: 0 (none)
          host timeout: 0 (none)
           window size: 4096 bytes
      fold program name: on
                 CCSID: 037
  number of connections: 0
  number of transactions: 0
           client type: CICS
```

d. If you defined any routes for specific transaction IDs to take to CICS destinations, enter the **show txconn route** command. Make sure that all CTRC routes you defined are listed with the configuration values you specified. A <default> in the SERVER column indicates a global route that can be used by all txconn servers on the router. A <default> in the TranID column indicates the default route for the listed txconn server.

```
Router# show txconn route
Server            TranID            Destination
----------------- ----------------- ----------------
CICSC             <default>         CICSC
CICSB             <default>         CICSB
CICSB&C           <default>         GEN
GUAVA             <default>         GUAVA
<default>         CPMI              CICSC
CICSB             CPMI              CICSB
```

Step 3 If you configured CTRC for communications with DB2, perform the following steps to verify the router is properly configured. If you are using CTRC only for CICS communications, skip to Step 4.

a. Enter the **show dbconn server** command. Make sure the servers you defined are listed with the configuration values you specified.

```
Router# show dbconn server
Server      Port   IPAddress       RDBName     State     NumConn
SERVERA     446    0.0.0.0         MATTY       enabled   0
SERVERB     446    0.0.0.0         SCU_DSNM    enabled   0
SERVERC     446    0.0.0.0         DSN4        enabled   0
SERVERD     446    0.0.0.0         MKTG        enabled   0
SERVERE     446    0.0.0.0         ABBY        enabled   0
SERVERF     446    0.0.0.0         DB2510      enabled   0
SERVERG     446    0.0.0.0         ELLE        enabled   0
SERVERH     446    0.0.0.0         SUNSET      enabled   0
SERVERI     446    0.0.0.0         NELL        enabled   0
SERVERJ     446    198.989.999.32  SAMPLE      enabled   0
SERVERK     446    0.0.0.0         DB2410      enabled   0
SERVERL     446    0.0.0.0         SQLDS       enabled   0
SERVERM     446    0.0.0.0         STELLA      enabled   0
SERVERN     446    10.10.19.4      OAK         enabled   0
SERVERO     447    0.0.0.0         DB2510      enabled   0
BUDDY       446    0.0.0.0         DB2510      enabled   0
```

Use the **show dbconn server** *server-name* form of the command to display more information for an individual server.

```
Router# show dbconn server BUDDY
                server: BUDDY
          server state: enabled (accepting connections)
            ip-address: 0.0.0.0
```

```
                port: 446
              rdbname: DB2510
      connection type: SNA
                 rlu: STARW.DSNV510
                mode: IBMRDB
              tpname: \x076DB
         idle-timeout: 0 (none)
          window-size: 4096 bytes
  database server name: (unknown)
   database product id: (unknown)
                 PEM: not configured
 number of connections: 0
           RDB server: active
                  WLM: inactive-enabled
```

b. For each dbconn server shown in the previous step, enter the **dbconn ping** command to verify that the router can communicate with the DB2 systems associated with that server.

```
Router# dbconn ping BUDDY
......
RDB named DB2510 on database server BUDDY successfully contacted!
Elapsed time was 00:00:00
```

Step 4 Verify that the CTRC license configuration matches the number of licensed connections that you purchased. Enter either the **show dbconn license** command or the **show txconn license** command as shown below.

```
Router# show txconn license

Router# show dbconn license
```

The command displays information about the license, as shown in the following example:

```
CTRC is licensed for 4990 connections, no licensed connections in use
This is a permanent license
```

Configuring CTRC Clients

This section provides information about setting up DRDA client connections for DB2 access, and for setting up the supported CICS clients.

Setting Up DB2 DRDA Client Connections

To configure a connection between a DRDA-based client and a DB2 database, you must define a data source to the ODBC driver. For each DB2 database that will be accessed, you need to specify the following data source information to configure the DRDA requestor to use the CTRC router:

- The RDB name of the DB2 database you want to access. This value must match the rdbname that you specify with the **dbconn server** command to configure the CTRC router for communicating with DB2 (see the "Configuring CTRC for DB2 Communications" section on page 16). The RDB name also must match the DDF location defined on the DB2 host (see the "Configuring DDF" section on page 11).
- The router's host name or the IP address of the interface that will accept the connection requests.
- The port number on which the CTRC router is listening for connection requests. The default is 446.

The procedures for configuring a data source are specific to the client implementation. Refer to the documentation for your DRDA client for details.

Setting Up CICS Clients

CTRC supports IBM CICS Universal Client, IBM TXSeries, and Microsoft COMTI clients. These clients connect to the Cisco router via TCP/IP.

Setting Up CICS Universal Client Connections

To set up the CICS Universal Client, perform the following tasks:

Step 1 Install the Universal Client for your platform.

Step 2 Choose TCP/IP as your network connection.

Step 3 To have the Universal Client connect to your CTRC server, add an entry in the Server section of the CICSCLI.INI file to define the CTRC server. The following example entry defines a server named CTRCSERV with a TCP/IP hostname (NetName) of CTRCBOX. Substitute the LU name of your router for the NetName.

```
Server = CTRCSERV
Description = TCP/IP Server
Protocol = TCPIP
NetName = CTRCBOX
Port = 1435
```

Step 4 If necessary, stop and restart the Universal Client to have the changes take effect and connect to the CTRC server.

To connect through multiple servers, increase the MaxServers value in the Client section of the CICSCLI.INI file from the default of 1. If you have multiple servers configured in CICSCLI.INI, some applications may display a list of servers from which to choose. If security is turned on in CICS, a user/password dialog may appear after selecting a CICS Server.

If you have specified UseDfltUser=NO and AttachSec=Verify in your APPC CONNECTION definition on CICS (see the "Defining APPC Connections to CTRC" section on page 9), a userid and password will be required to use the CICS Terminal. If you are using ECI, pass the userid and password using a command such as:

```
cicscli /c=ctrcserv /u=p390 /p=p390
```

The CICS Terminal status line displays the virtual terminal name. When you enter a command on the terminal (such as "CEOT"), you will see the SYSID and APPLID of the CICS system to which you are connected.

Setting Up TXSeries as a CTRC Client

To connect a machine running TXSeries to another CICS host through a CTRC connection, you must create the following CICS resource definitions:

- Listener Definition
- Communications Definition
- Program Definition for each remote program you want to use

You can create these resource definitions using the **cicsadd** command, or you can use the CICS System Management Interface Tool (SMIT) to build the commands. The following sections describe both methods.

Note The procedures in the following sections show how to create the resource definitions for TXSeries on AIX. If you are using TXSeries on Windows NT, refer to your TXSeries documentation for the commands and configuration panels provided for creating resource definitions on that platform.

Using cicsadd to Create the Definitions

To use the **cicsadd** command to add CICS resource definitions on TXSeries for AIX, specify the values appropriate for your definition in place of the variables shown in italic in the following command syntax.

```
cicsadd -c className[-r regionName] [-P | -B] [-f fileName] [-m modelId] resourceName
[attributeName=attributeValue ...]
```

To use the CTRC router, the value for the *resourceName* in the Communications Definition (CD) must be the same as the *attributeValue* specified for the RemoteSysId attribute in the Program Definition. And, the ListenerName specified in the CD must match the name of the Listener Definition. For example, issuing the following command creates a Communications Definition for the CTRC router with a *resourceName* of CTRC and a ListenerName of TCP:

```
cicsadd -c cd -r TX6000 -B CTRC ResourceDescription="Connection thru CTRC"
ConnectionType=cics_tcp ListenerName=TCP OutboundUserIds=sent RemoteCodePageTR="IBM-037"
RemoteNetworkName="CICSB" RemoteSysSecurity=trusted RemoteTCPAddress="ctrcbox"
RemoteTCPPort=1435 RemoteLUName="CTRCBOX"
```

To use a remote program named PNG1, the Program Definition for PNG1 must set the RemoteSysId attribute to CTRC, as shown in the following command.

```
cicsadd -c pd -r TX6000 -B PNG1 ResourceDescription=”eciPing back end” RemoteSysId=CTRC
RemoteName=PNG1 RSLKey=public
```

You specify the protocol that the CICS client will use in the Listener Definition. For example, to allow the TXSeries client to connect to the CICS region specified in the above example commands, TX6000, add a Listener Definition for TCP/IP as shown in the following command.

```
cicsadd -c ld -r TX6000 -B TCP ResourceDescription=”TCP/IP Listener” Protocol=TCP
```

Using SMIT to Create the Definitions

To use SMIT to build the commands for creating the resource definitions, start SMIT and display the Manage Resources menu, which lists the types of definitions you can create.

Following are example definitions, assuming the values below for the CTRC-related parameters:

- TX6000—Name of the CICS region on an RS/6000 running TXSeries.
- CTRCBOX—IP host name of CTRC router.
- CICSB—APPLID of CICS server running on a mainframe.
- PNG1—ECI host program running on the mainframe.

Listener Definition Example

```
* New Listener Identifier                    [TCP]
* Listener Identifier                         TCP
* Region name                                 TX6000
  Update Permanent Database OR
       Install OR Both                       Both
  Group to which resource belongs            []
  Activate resource at cold start?            yes
  Resource description                       [Listener Definition]
* Number of updates                           0
  Protect resource from modification?         no
  Protocol type                               TCP
  TCP adapter address                        [198.147.235.8]
  TCP service name                           []
  local SNA Server Protocol Type              TCP
  local SNA Server Identifier                []
  local SNA Node Name                        []
  local Named Pipe name                      []
```

Communication Definition Example

The following definition shows a TCP/IP link to a CICS host STARW.CICSB through the CTRC router named CTRCBOX:

```
  New Communication Identifier                    [CTRC]
  Communication Identifier                         CTRC
  Region name                                      TX6000
  Update Permanent Database OR
        Install OR Both                           Both
  Group to which resource belongs                 []
  Activate the resource at cold start?             yes
  Resource description                            [Communications Definit>
* Number of updates                                2
  Protect resource from modification?              no
  Connection type                                  cics_tcp
  Name of remote system                           [CICSB]
  SNA network name for the remote system          [STARW]
  SNA profile describing the remote system        []
  Default modename for a SNA connection           []
  Gateway Definition (GD) entry name              []
  Listener Definition (LD) entry name             [TCP]
  TCP address for the remote system               [CTRCBOX]
  TCP port number for the remote system           [1435]
  DCE cell name of remote system                  [/.:/]
  Timeout on allocate (in seconds)                [60]
  Code page for transaction routing               [IBM-037]
  Set connection in service?                       yes
  Send userids on outbound requests?               sent
  Security level for inbound requests              verify
  UserId for inbound requests                     []
  Transaction Security Level (TSL) Key Mask       [none]
  Resource Security Level (RSL) Key Mask          [none]
  Transmission encryption level                    none
```

Program Definition Example

The following definition describes a program named PNG1 that is running on the remote system accessed through the Communication Definition named CTRC (see the "Communication Definition Example" section on page 24):

```
  New Program Identifier                           [PNG1]
  Program Identifier                                PNG1
  Region name                                       TX6000
  Update Permanent Database OR Install
          OR Both                                  Both
  Group to which resource belongs                  []
  Activate resource at cold start?                  yes
  Resource description                             [Program Definition]
* Number of updates                                 0
  Protect resource from modifications?             no
  Program enable status                             enabled
  Remote system on which to run program            [CTRC]
  Name to use for program on remote system         [PNG1]
  Transaction name on remote system for program    []
  Resource Level Security Key                      [public]
  Program path name                                []
  Program type program
  User Exit number                                 [0]
  Is a user conversion template defined?            no
  Is this a program that should be cached?          no
```

Refer to the IBM TXSeries CICS documentation for more information about specifying CICS resource definitions on TXSeries.

Setting Up COMTI Client Connections

When a COMTI application is built using Microsoft's COMTI Component Builder, it must be defined with the following information to provide remote access to CICS.

- "CICS and IMS via TCP/IP" as the remote environment type
- "CICS" as the target environment
- "MS Link" as the server mode

For the COMTI client to access CICS using the CTRC router, you must define CTRC as a TCP Remote Environment. Use Miscrosoft's COMTI Manager to define the remote environment with the following values.

- Select "CICS and IMS using TCP/IP" as the remote environment type
- Specify the IP address and TCP port address as configured on the CTRC router
- Specify a name and comment for the new remote environment

Refer to the *Microsoft COM Transaction Integrator Online Guide* for details about setting up and using COMTI.

Monitoring and Maintaining CTRC

This section describes commands used to monitor and maintain CTRC. Commands for CICS communications and DB2 communications are shown separately.

Note CTRC commands related to communications with CICS contain the word **txconn**. CTRC commands related to communications with DB2 contain the word **dbconn**. With the exception of commands related to licensing, **dbconn** and **txconn** commands act independently of each other.

Monitoring and Maintaining CTRC Communications with CICS

To monitor and maintain CTRC communications with CICS, use the following commands in privileged EXEC mode:

Command	Purpose
Router# **clear txconn connection** *connection-id*	Terminates the specified CTRC connection to a CICS client and all associated transactions.
Router# **clear txconn statistic name** {**allocatetime** | **clientreceived** | **clientsent** | **clientturnaround** | **every** | **hostreceived** | **hostresponse** | **hostsent** | **maxconnections** | **maxtransactions** | **totalconnections** | **totaltransactions**}	Clears the named statistic or all statistics (**every** keyword) related to CTRC communications with CICS.
Router# **clear txconn transaction** *transaction-id*	Terminates the specified CTRC transaction.
Router# **debug txconn** {**all** | **appc** | **config** | **data** | **event** | **tcp** | **timer**}	Enables debugging of CTRC communications with CICS.
Router# **show debugging**	Displays current status of debugging for the router.

Command	Purpose
Router# **show txconn connection** [**server** *server-name*]	Displays a list of all CTRC connections to CICS clients from the current router, or a particular server's CICS client connections.
Router# **show txconn connection** *connection-id*	Displays detailed status information for the specified CTRC connection to a CICS client.
Router# **show txconn destination** *destination-name*	Displays a list of all the current router's destinations for CICS communications, or detailed status information for the specified CTRC destination.
Router# **show txconn license** or **show dbconn license**	Shows the status of the CTRC license.
Router# **show txconn route** [**server** *server-name*]	Displays a list of CTRC routes to CICS for the current router or a particular server.
Router# **show txconn server**	Lists the CTRC servers that are configured for CICS communications on the current router.
Router# **show txconn server** *server-name*	Displays detailed status information for the specified CTRC server.
Router# **show txconn transaction** [**server** *server-name* | c**onnection** *connection-id*]	Displays a list of the current router's CTRC transactions with CICS, or the transactions of a particular server or connection.
Router# **show txconn transaction** *transaction-id*	Displays detailed status information for the specified CTRC transaction.
Router# **show txconn statistic** [**kind** {**histogram** | **summary**}] **name** {**activeconnections** | **activetransactions** | **allocatetime** | **clientreceived** | **clientsent** | **clientturnaround** | **dump** | **hostreceived** | **hostresponse** | **hostsent** | **latency** | **maxconnections** | **maxtransactions** | **totalconnections** | **totaltransactions**}	Displays statistics related to CTRC communications with CICS.
Router# **txconn ping** *destination-name*	Tests communications between the CTRC router and a CTRC destination (a host defined by a pair of RLU and mode values).

Monitoring and Maintaining CTRC Communications with DB2

To monitor and maintain CTRC communications with DB2, use the following commands in privileged EXEC mode:

Command	Purpose
Router# **clear dbconn connection** *connection-id*	Breaks the specified client connection to the server.
Router# **clear dbconn statistic** {**chains** | **clientturnaround** | **connectionsdown** | **connectionsup** | **every** | **hostreceived** | **hostresponse** | **hostsent** | **maxconnections**}	Clears statistics related to CTRC communications with DB2.
Router# **dbconn ping** *server-name* [**userid** *userid*] [**password** *password*] [**rdbname** *rdbname*]	Verifies connectivity to the specified DB2 database.
Router# **debug dbconn** {**all** | a**ppc** | **config** | **drda** | **event** | **tcp**}	Enables debugging of CTRC communications with DB2.

Command	Purpose
`Router# show dbconn connection`	Displays the status of each CTRC connection to DB2.
`Router# show dbconn connection` *connection-id*	Displays a detailed status of the specified CTRC connection to DB2.
`Router# show dbconn connection server` *server-name*	Displays the status of CTRC connections to DB2 for the specified server.
`Router# show dbconn connection userid` *userid*	Displays the status of a user connected to CTRC for DB2 communications.
`Router# show dbconn connection rdbname` *rdb-name*	Displays a status of each connection to DB2 that matches the specified RDB name.

`Router# show dbconn license`
or
`Router# show txconn license`
	Displays the status of the CTRC license for both DB2 and CICS.

`Router# show dbconn ports`	Displays information on all ports through which CTRC servers are accepting connections to DB2.
`Router# show dbconn server`	Displays a summary of information about each CTRC server configured to communicate with DB2.
`Router# show dbconn server` *server-name*	Displays a detailed status of the specified CTRC server for DB2 communications.
`Router# show dbconn statistic [kind {histogram | summary}] name {chains | clientturnaround | connectionsdown | connectionsup | dump | hostreceived | hostresponse | hostsent | latency | maxconnections}`	Displays current statistics related to CTRC communications with DB2.
`Router# show debugging`	Displays current status of debugging for CTRC.

CTRC Configuration Examples

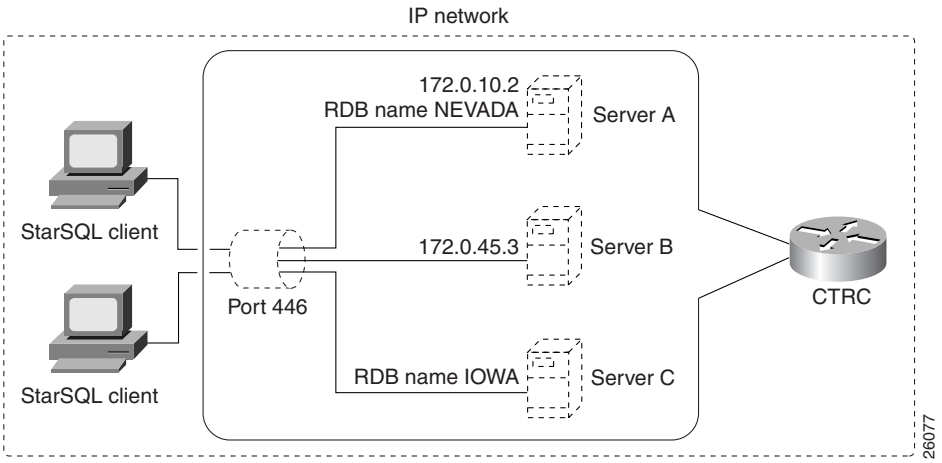
The following sections provide CTRC configuration examples:

CTRC Servers with IP Addresses Configuration Example (DB2)

Figure 240 shows a CTRC configuration where the CTRC servers are configured to listen on port 446 (by default) for IP addresses specified for these servers in the router's configuration for CTRC. When an ODBC client attempts to make a connection to DB2, a CTRC server accepts the connection if the IP address specified in its configuration matches the IP address to which the client wants to connect.

In this illustration, Servers A and B are configured with IP addresses 172.0.10.2 and 172.0.45.3. Servers A and B accept any connection that targets their IP addresses. Server C accepts any connection that targets any IP address of router on the target port of 446 and an RDB name of IOWA.

Figure 240 CTRC Servers' Configuration with IP Addresses (for DB2 Communications)



The following are the commands that configure Server A, Server B, and Server C in the Cisco router:

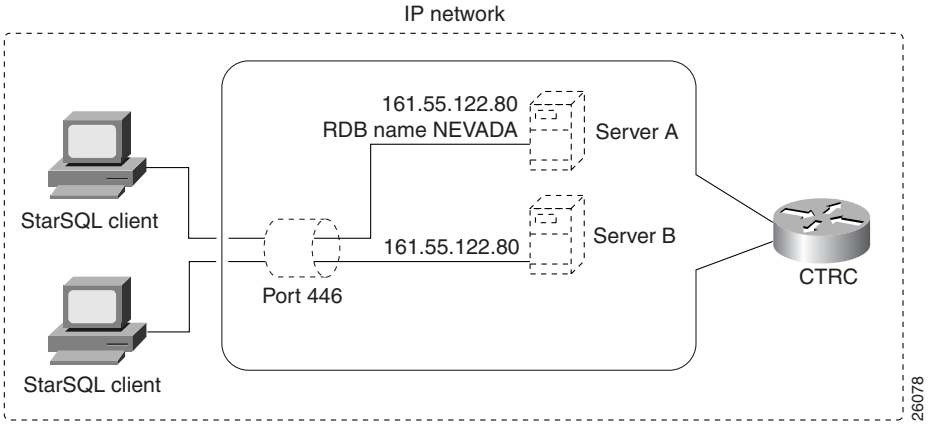
```
dbconn server SERVERA ip-address 172.0.10.2 rdbname nevada
dbconn server SERVERB ip-address 172.0.45.3
dbconn server SERVERC rdbname iowa
```

CTRC Servers with IP Addresses, RDB Names, and Ports Configuration Example 1 (DB2)

When a client request comes in for a server, and multiple servers are configured in the router, the three configured attributes of IP address, RDB name, and port determine which server is chosen for the connection. When a server is selected for a connection, the client remains associated with that server for the duration of that connection. The APPC attributes configured for that server are used to connect to the IBM system. If a server is unconfigured while active connections exist, the active connections with that server will break.

Only one CTRC server can be configured with a unique combination of IP address, port, and RDB name. If a situation arises where multiple servers in a router meet the criteria for accepting a client connection, the CTRC server that meets the most specific criteria accepts the connection. For example, in Figure 241 Servers A and B are listening on port 446 for client connections that match their IP address of 161.55.122.80. Server A is configured to accept RDB name NEVADA and Server B is configured to accept any RDB name. A client connecting to port 446 for RDB name NEVADA matches the criteria for both servers. In this situation, Server A is selected to accept the connection because its configuration includes a specific RDB name NEVADA as compared to Server B whose configuration accepts any RDB name.

Figure 241 CTRC Server Configuration with IP Address and RDB Name Defined



CTRC Servers with IP Addresses, RDB Names, and Ports Configuration Example 2 (DB2)

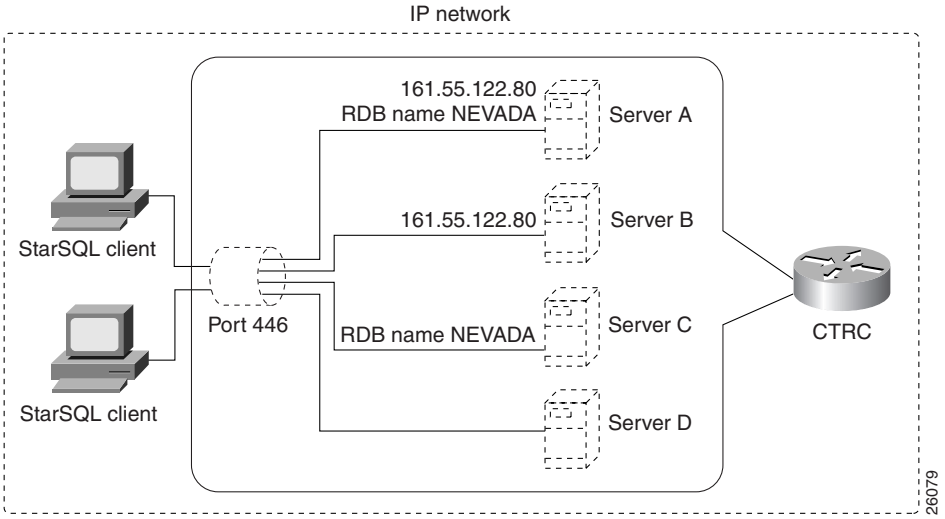
The IP address and port specified for a server in a router's configuration also determines which server accepts a connection. For example, Server C is configured to listen on any local IP address on port 446 and RDB name IOWA. Server D is configured to listen for IP address 145.56.180.34 on port 446 and RDB name IOWA. When a client attempts to connect to IP address 145.56.180.34 on port 446 for RDB name IOWA, both servers meet the criteria in accepting the connection. In this case, CTRC selects a connection based on the IP address first, then the port, and finally, the RDB name.

Server Selection by IP Addresses, RDB Names, and Ports Configuration Example (DB2)

If multiple servers in a router meet the criteria for accepting a client connection, the CTRC server that meets the most specific criteria accepts the connection. In Figure 242, the Cisco router contains four server configurations. All four servers listen for client connections on port 446 by default. Both Servers A and B are configured with the same IP address, 161.55.122.80. Servers A and C are configured to accept RDB name NEVADA. Servers B and D are configured to accept any RDB name.

If a client connects to IP address 161.55.122.80 on port 446 and sends RDB name NEVADA in the DRDA data stream, all four servers match the criteria for accepting the client connection. However, Server A will be selected to accept the connection because it meets the most specific criteria for IP address, RDB name, and port. If Server A was not configured, Server B would be the second choice because it meets the criteria for the IP address and port. The IP address specified in a server always has precedence when matching a connection to a server.

Figure 242 CTRC Server Configurations with IP Addresses, RDB Names, and Default Port



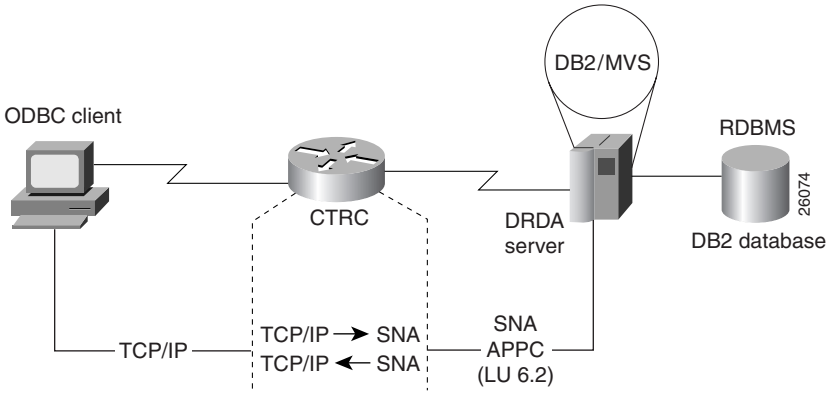
The following is the configuration for Servers A, B, C, and D in the Cisco router:

```
hostname routera
!
enable password allie

dbconn server SERVERA ip-address 161.55.122.80 rdbname NEVADA
dbconn server SERVERB ip-address 161.55.122.80
dbconn server SERVERC rdbname NEVADA
dbconn server SERVERD
```

CTRC with CIP and DB2 on VTAM Configuration Example (DB2)

Figure 243 illustrates a Cisco router with a CIP that is configured with CTRC. The CIP is networked and connected to VTAM on the mainframe. DB2 is configured on VTAM.

Figure 243 Cisco Router with CIP and Connection to DB2 on VTAM



The configuration in Figure 243 uses router commands to configure SNA Switching Services over CIP and CSNA via SRB. The following examples show the configuration in more detail.

In the VTAM host definitions, the variable CONNTYPE=APPN is optional, but is recommended if you use APPN in your SNA environment. If CP-to-CP is set to YES and CONNTYPE is set to APPN, this configuration enables the Cisco router to establish CP-to-CP sessions with VTAM. By allowing CP-to-CP sessions, you gain the benefit of APPN's dynamic features such as the availability of directory and topology for locating resources and calculating optimal routes.

VTAM Partner PU and LU Definition

```
CTRCPU PU ADDR=01,                                    X
          IDBLK=05D,                                  X
          IDNUM=00501,                                X
          CPNAME=CTRCBOX,                             X
          ANS=CONT,                                   X
          DISCNT=NO,                                  X
          IRETRY=NO,                                  X
          ISTATUS=ACTIVE,                             X
          PUTYPE=2,                                   X
          SECNET=NO,                                  X
          MAXDATA=521,                                X
          MAXOUT=7,                                   X
          MAXPATH=1,                                  X
          USSTAB=USSS,                                X
          MODETAB=ISTINCLM,                           X
          DLOGMOD=IBMRDB,                             X
          CONNTYPE=APPN
CTRCBOX LU  LOCADDR=00,       INDEPENDENT LU          X
            DLOGMOD=IBMRDB
```

VTAM APPLID for DB2

```
DSNV510 APPL   APPC=YES,                             X00006012
               AUTH=ACQ,                             X00007012
               AUTOSES=1,                            X00008012
               DMINWNL=1024,                         X00009012
               DMINWNR=1024,                         X00009112
               DSESLIM=2048,                         X00009212
               EAS=65535,                            X00009312
               MODETAB=ISTINCLM,                     X00009412
               SECACPT=CONV,                         X00009512
               SRBEXIT=YES,                          X00009612
               VERIFY=NONE,                          X00009712
               VPACING=1,                            X00009812
               SYNCLVL=SYNCPT,                       X00009912
               ATNLOSS=ALL                            00010012
```

XCA for a CIP-Attached Router

```
XCAE20    VBUILD  TYPE=XCA
XPE20R    PORT    CUADDR=E20,
                  ADAPNO=1,
                  SAPADDR=4,
                  MEDIUM=RING,
                  DELAY=0,
                  TIMER=60
G02E20A  GROUP ANSWER=ON,CALL=INOUT,DIAL=YES,ISTATUS=ACTIVE
K02T201S  LINE
P02T201S    PU
K02T202S  LINE
P02T202S    PU
```

Cisco IOS Software Configuration

In this example, the router CTRCBOX is attached to the host BUDDY using a CIP processor. Note that the source-bridge ring-group of 100 matches the source bridge of 10 2 100 for interface Channel 13/2 to enable SNA Switching Services to run over SRB. In addition, the destination LAN address used by the SNASw link station BUDDY corresponds to the virtual MAC address used by the adapter for Channel 13/2.

```
!
source-bridge ring-group 100
!
interface Ethernet2/1
 mac-address 4200.0000.0501
 ip address 198.147.235.11 255.255.255.0
 no ip directed-broadcast
 no ip mroute-cache
!
interface Channel3/0
 ip address 192.168.1.1 255.255.255.0
 no ip directed-broadcast
 no keepalive
 channel-protocol S4
 claw 0100 22 192.168.1.2 BUDDY CIPTCP TCPIP TCPIP
 csna 0100 20
!
interface Channel3/2
 no ip address
 no ip directed-broadcast
 no keepalive
 lan TokenRing 1
  source-bridge 10 2 100
  adapter 1 4000.0123.9999
!
interface Virtual-TokenRing0
  mac-address 4000.2222.3333
  source-bridge 50 1 100
  source-bridge spanning
!
snasw cpname STARW.CTRCBOX
snasw port SRB Virtual-TokenRing0
snasw link BUDDY port SRB rmac 4000.0123.9999
snasw location DSNV510 owning-cp STARW.BUDDY (see Note below)

!
dbconn server DB2BUDD rdbname DB2510 rlu STARW.DSNV510 mode IBMRDB
!
ip default-gateway 198.147.235.12
ip classless
```

Note Do not use an **snasw location** statement if you are using an APPN connection between the host and SNASw.

CTRC Servers Using Token Ring to a LEN Configuration Example (CICS and DB2)

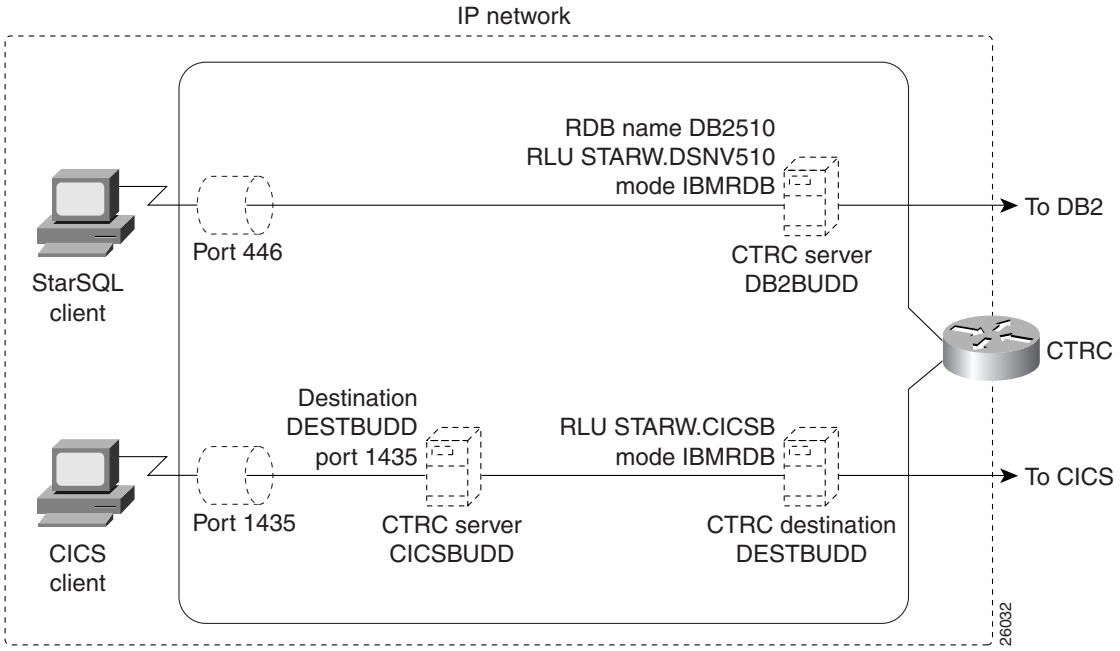
This section provides a configuration example for a router named CTRCBOX, beginning with the VTAM definition for the router, which is the same as for the previous example.

The router is connected to the host via Token Ring. The control point name of the host is BUDDY; its Token Ring MAC address is 4000.0200.0448.

The host is configured as a Subarea Node (APPN LEN); if a host is configured as an APPN Network Node, the SNASw location statements are unnecessary.

Figure 244 shows a CTRC configuration for communication with DB2 and CICS.

Figure 244 CTRC Configuration for Communication with DB2 and CICS



VTAM Partner PU and LU Definition

```
CTRCPU  PU ADDR=01,                                    X
           IDBLK=05D,                                  X
           IDNUM=00501,                                X
           CPNAME=CTRCBOX,                             X
           ANS=CONT,                                   X
           DISCNT=NO,                                  X
           IRETRY=NO,                                  X
           ISTATUS=ACTIVE,                             X
           PUTYPE=2,                                   X
           SECNET=NO,                                  X
           MAXDATA=521,                                X
           MAXOUT=7,                                   X
           MAXPATH=1,                                  X
           USSTAB=USSS,                                X
           MODETAB=ISTINCLM,                           X
           DLOGMOD=IBMRDB,                             X
           ONNTYPE=APPN
CTRCBOX LU  LOCADDR=00,       INDEPENDENT LU           X
            DLOGMOD=IBMRDB
```

VTAM APPLID for CICS

```
CICSAPPL VBUILD TYPE=APPL                                        00010001
*************************************************************** 00010000
* CICS APPL DEFINITION FOR LU62 CLIENT/SERVER SUPPORT           00020000
*************************************************************** 00030000
CICSB    APPL  AUTH=(ACQ,SPO,PASS,VPACE),                       X
         MODETAB=ISTINCLM,                                      X
         VPACING=0,EAS=100,PARSESS=YES,                         X
         APPC=NO,                                               X
         SONSCIP=YES,                                           X
         ACBNAME=CICSB
```

VTAM APPLID for DB2

```
DSNV510 APPL   APPC=YES,                          X00006012
               AUTH=ACQ,                          X00007012
               AUTOSES=1,                         X00008012
               DMINWNL=1024,                      X00009012
               DMINWNR=1024,                      X00009112
               DSESLIM=2048,                      X00009212
               EAS=65535,                         X00009312
               MODETAB=ISTINCLM,                  X00009412
               SECACPT=CONV,                      X00009512
               SRBEXIT=YES,                       X00009612
               VERIFY=NONE,                       X00009712
               VPACING=1,                         X00009812
               SYNCLVL=SYNCPT,                    X00009912
               ATNLOSS=ALL                         00010012
```

VTAM APPLID for PEM Support

```
MVSLU01     APPL       ACBNAME=MVSLU01,     ACBNAME FOR APPC
                       APPC=YES,
                       AUTOSES=0,
                       DDRAINL=NALLOW,
                       DLOGMOD=IBMRDB,
                       DMINWNL=5,
                       DMINWNR=5,
                       DRESPL=NALLOW,
                       DSESLIM=10,
                       LMDENT=19,
                       PARSESS=YES,
                       SECACPT=CONV,
                       SRBEXIT=YES,
                       VPACING=1
```

DB2 BSDS DDF Record

The following example updates the BSDS with a location name of DB2510, LU name of DSNV510 for SNA access, a password of STARPASS, and a port of 446 for TCP/IP communications. The RESPORT and PORT parameters are required only for TCP/IP access and can be omitted if using only SNA.

```
//*
//DSNTLOG  EXEC  PGM=DSNJU003,COND=(4,LT)
//STEPLIB   DD   DISP=SHR,DSN=DSN510.SDSNLOAD
//SYSUT1    DD   DISP=OLD,DSN=DSN5CAT.BSDS01
//SYSUT2    DD   DISP=OLD,DSN=DSN5CAT.BSDS02
//SYSPRINT  DD   SYSOUT=*
//SYSUDUMP  DD   SYSOUT=*
//SYSIN     DD   *
            DDF  LOCATION=DB2510,LUNAME=DSNV510,
                 PASSWORD=STARPASS,RESPORT=5020,PORT=446
//*
```

XCA for Token Ring Attached Router

```
XCAE40    VBUILD  TYPE=XCA
XPE40R    PORT    CUADDR=E40,
                  ADAPNO=1,
                  SAPADDR=4,
                  MEDIUM=RING,
                  DELAY=0,
                  TIMER=30
G02E40A   GROUP DIAL=YES,CALL=INOUT,ANSWER=ON,ISTATUS=ACTIVE
*
K02T001S  LINE
P02T001S    PU
*
K02T002S  LINE
P02T002S    PU
```

Cisco IOS Software Configuration

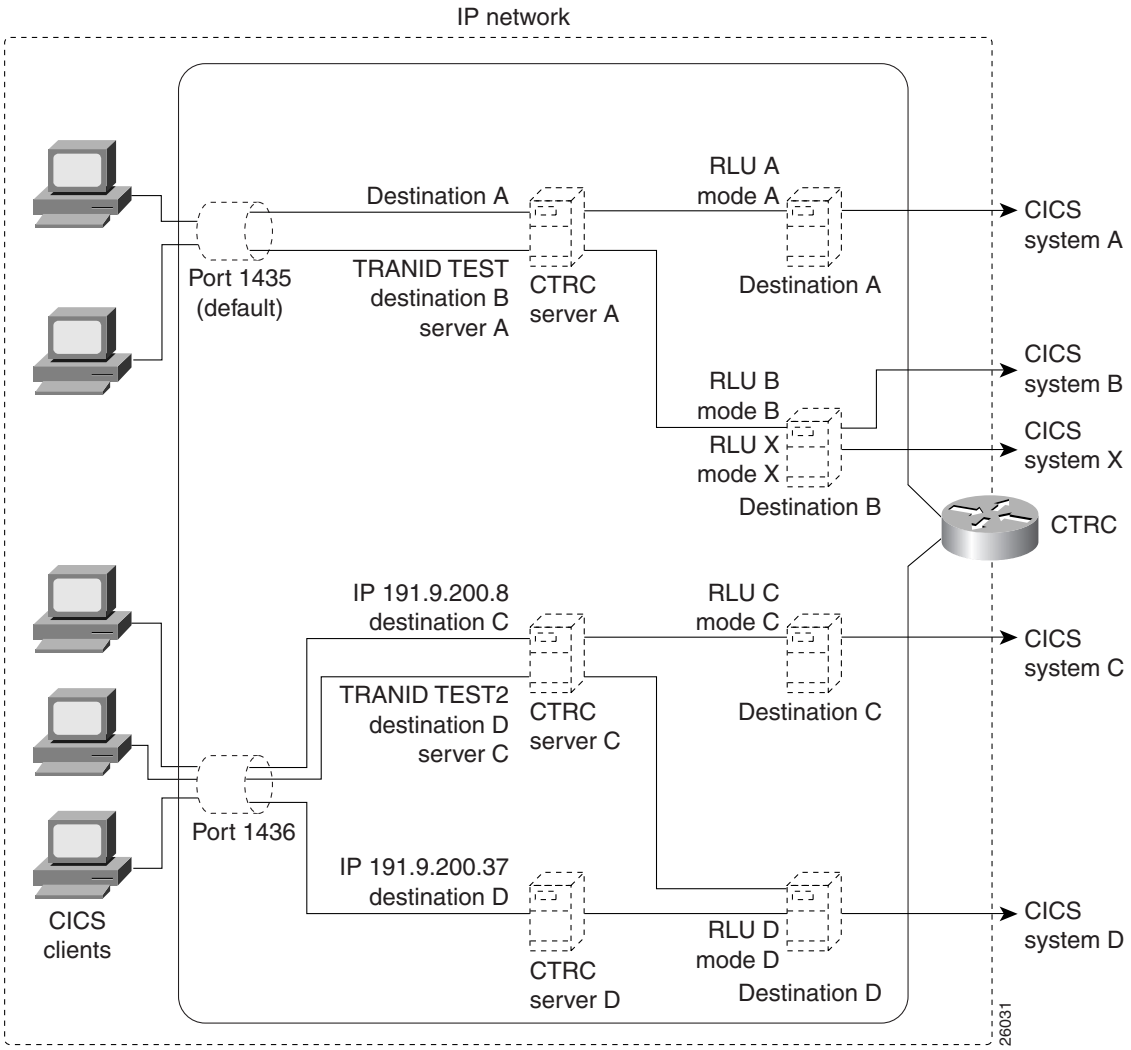
```
source-bridge ring-group 100
!
!
interface TokenRing0/1
 mac-address 4000.1111.0501
 ip address 198.147.236.196 255.255.255.0
 no ip directed-broadcast
 no ip mroute-cache
 early-token-release
 ring-speed 16
 multiring all
!
interface Ethernet2/1
 mac-address 4200.0000.0501
 ip address 198.147.235.11 255.255.255.0
 no ip directed-broadcast
 no ip mroute-cache
!
!
snasw cpname STARW.CTRCBOX
snasw port TR0 TokenRing0/1
snasw link BUDDY port TR0 rmac 4000.0200.0448
snasw location STARW.DSNV510 owning-cp STARW.BUDDY
snasw location STARW.CICSB owning-cp STARW.BUDDY
!
dbconn server DB2BUDD rdbname DB2510 rlu STARW.DSNV510 mode IBMRDB
dbconn tcpserver BUDDTCP port 446 rdbname DB2510 remote-ip-address 198.147.235.39
remote-port 446
        dbconn pem DB2BUDD rlu MVSLU01 mode #INTER
!
txconn destination DESTBUDD rlu STARW.CICSB mode IBMRDB
txconn server CICSBUDD destination DESTBUDD port 1435
ip default-gateway 198.147.235.12
ip classless
```

CTRC Servers with IP Addresses, Routes, and Multi-Valued Destinations Configuration Example (CICS)

Figure 245 shows a CTRC configuration that includes multiple CTRC servers, routes, default and non-default ports, and one multi-valued CTRC destination. This example illustrates the following CTRC configuration principles:

- One router can run multiple CTRC **txconn** servers.
- One **txconn** server can communicate with multiple logical destinations.
- One CTRC logical destination can correspond to multiple CICS destination systems.
- More than one **txconn** server can use a single port number, provided that each server listens on a different IP address.
- More than one **txconn** server can direct traffic to a single logical destination.

Figure 245 CTRC Configuration with IP Addresses, Routes, and Multiple CICS Destinations



In Figure 245, a single router is configured to run three CTRC servers for communication with CICS. These **txconn servers** are shown as CTRC server A, CTRC server C, and CTRC server D. Server A listens on the default port, 1435, for all of the router's IP addresses. Server C listens on port 1436 for IP address 191.9.200.8. Server D listens on port 1436 for IP address 191.9.200.37.

Server A is configured to communicate with two logical destinations. If a client communication has the value of TEST for its transaction ID (TRANID), server A sends it to logical Destination B. This is a multi-valued destination that allows communication with two CICS systems, system B (with RLU B and mode B) and system X (with RLU X and mode X). CTRC allocates transactions to these two destination systems on a round-robin basis.

If a client communication for server A does not have a value of TEST for TRANID, server A sends it to Destination A, which corresponds to CICS system A (with RLU A and mode A).

Server C is also configured to communicate with two logical destinations. If server C receives a client communication that has the value of TEST2 for its transaction ID, server C sends it to logical Destination D, which corresponds to CICS system D (with RLU D and mode D). Server C sends client communications with other transaction IDs to logical Destination C (CICS system C, with RLU C and mode C). Server D is configured to send client communications to logical Destination D.

Cisco IOS Software Feature Removal: CPP, CMPC, CTRC, LNM, TR-ISL, and TR-LANE

Feature History

Release	Modification
12.3(2)T	Features described in this feature module were removed in Cisco IOS Release 12.3(2)T.
12.3(4)T	Features described in this feature module were removed in Cisco IOS Release 12.3(4)T.

The Cisco IOS Software Feature Removal feature is an engineering project to permanently remove selected legacy features (or components) from the Cisco IOS code. These features will not be available in future releases of Cisco IOS software.

The legacy feature that has been removed as of Release 12.3(2)T is as follows:

- Token Ring Inter-Switch Link

The legacy feature that has been removed as of Release 12.3(4)T are as follows:

- Combinet Packet Protocol
- Cisco Multipath Channel
- Cisco Transaction Connection
- LAN Network Manager
- Token Ring LAN Emulation

This document lists the commands that have been removed from or modified in Cisco IOS software with the removal of a specified feature.

Note Commands that have been modified may not all be listed in this document.

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

Token Ring Inter-Switch Link

The following commands have been removed from Cisco IOS software with the removal of the Token Ring Inter-Switch Link (TRISL) feature. Please note that not all commands that may have been modified are listed.

- **encapsulation tr-isl trbrf-vlan**
- **multiring trcrf-vlan**
- **source-bridge trcrf-vlan**

Combinet Packet Protocol

The following commands have been removed from Cisco IOS software with the removal of the Combinet Packet Protocol (CPP) feature. Please note that not all commands that may have been modified are listed.

- **cpp authentication**
- **cpp callback accept**
- **encapsulation cpp**
- **debug cpp negotiation**
- **debug cpp packet**
- **debug cpp event**

Cisco Multipath Channel

The following commands have been removed from Cisco IOS software with the removal of the Cisco Multipath Channel (CMPC) feature. Please note that not all commands that may have been modified are listed.

- **cmpc**
- **tg (CMPC)**

Cisco Transaction Connection

The following commands have been removed from Cisco IOS software with the removal of the Cisco Transaction Connection (CTRC) feature. Please note that not all commands that may have been modified are listed.

- **clear dbconn connection**
- **clear dbconn statistic**
- **clear txconn connection**
- **clear txconn statistic**
- **clear txconn transaction**
- **dbconn license**
- **dbconn pem**
- **dbconn ping**

- **dbconn server**
- **dbconn tcpserver**
- **show dbconn connection**
- **show dbconn license**
- **show dbconn ports**
- **show dbconn server**
- **show dbconn statistic**
- **show dbconn wlm**
- **show txconn connection**
- **show txconn destination**
- **show txconn license**
- **show txconn route**
- **show txconn server**
- **show txconn statistic**
- **show txconn transaction**
- **txconn destination**
- **txconn license**
- **txconn ping**
- **txconn route**
- **txconn server**

LAN Network Manager

The following commands have been removed from Cisco IOS software with the removal of the LAN Network Manager (LNM) feature. Please note that not all commands that may have been modified are listed.

- **lnm alternate**
- **lnm crs**
- **lnm disabled**
- **lnm express-buffer**
- **lnm loss-threshold**
- **lnm password**
- **lnm pathtrace-disabled**
- **lnm rem**
- **lnm rps**
- **lnm snmp-only**
- **lnm softerr**
- **show lnm bridge**

- **show lnm config**
- **show lnm interface**
- **show lnm ring**
- **show lnm station**

Token Ring LAN Emulation

The following commands have been modified in Cisco IOS software with the removal of the Token Ring LAN Emulation (TR-LANE) feature. Please note that not all commands that may have been modified are listed.

- **lane client**
- **lane server-bus**

Configuring Cisco Mainframe Channel Connection Adapters

This chapter provides an introduction to the Cisco Mainframe Channel Connection Adapters (CMCCs) and provides information about the basic tasks required to configure any CMCC adapter on a Cisco router. This information is described in the following sections:

Details about configuring the Cisco IOS features that are supported by the CMCCs are described in the related chapters of this publication. For more information about the functions supported on a CMCC, see the "Supported Environments" section on page 6.

For hardware technical descriptions and information about installing the router interfaces, refer to the hardware installation and maintenance publication for your product. For a complete description of the CMCC adapter commands in this chapter, refer to the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Overview of the CMCC Adapters

A CMCC adapter is installed in a Cisco router to provide IBM channel attachment from the router to a mainframe host. The Cisco family of CMCC adapters consists of two basic types of adapters:

- Channel Interface Processor (CIP)—Installed on Cisco 7000 with RSP7000 and Cisco 7500 series routers
- Channel Port Adapter (CPA)—Installed on Cisco 7200 series routers

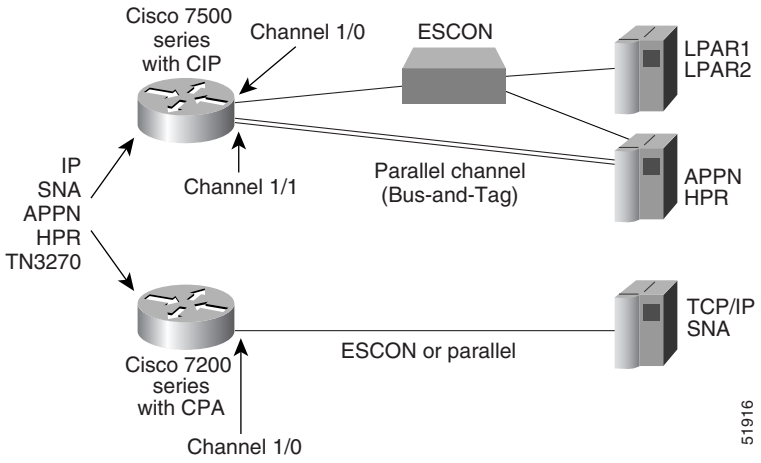
Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Each type of adapter (CIP or CPA) supports both ESCON and parallel channel attachment to the host and can eliminate the need for a separate front-end processor (FEP).

All CMCC adapters support the full range of channel software applications available in the Cisco IOS software including support for the Common Link Access to Workstation (CLAW) protocol, TCP/IP offload, IP host backup, Cisco SNA (CSNA), Cisco Multipath Channel (CMPC), Cisco Multipath Channel+ (CMPC+), and the TN3270 server.

Figure 246 shows the type of channel connections and environments supported by the CMCC adapters.

Figure 246 Cisco Mainframe Channel Connection Adapters



The following topics in this section provide additional overview information about the CMCC adapters:

- Channel Interface Processor, page 3
- Channel Port Adapter, page 4
- Differences between the CIP and CPA, page 5
- Supported Environments, page 6

Channel Interface Processor

The CIP for the Cisco 7000 with RSP7000 and Cisco 7500 series routers is designed for high-end network environments that demand high-performance, high-port density, and high-capacity solutions.

The CIP provides support for IBM ESCON and bus-and-tag parallel channel attachment using the following types of interfaces:

- ESCON Channel Adapter (ECA)
- Parallel Channel Adapter (PCA)

A single CIP can support up to two physical channel interfaces in any combination of either PCA or ECA. Each CIP is pre-configured with the appropriate channel adapters at manufacturing time.

The Cisco 7000 with RSP7000 and Cisco 7500 series routers support online insertion and removal (OIR), which allows you to install or remove CIPs while the system is operating.

Benefits of the CIP

The CIP provides the following primary benefits:

- Maximum throughput for every application—For the individual applications supported on the CIP, the CIP configured with 128 MB of memory offers maximum throughput. For example, the number of users supported for TCP/IP offload is 10,000 and the number of LLC2 session supported is 6000.
- Scalability—The CIP supports up to 22 channel connections on Cisco 7000 with RSP7000 and Cisco 7500 series routers.
- Multiple interface support—The CIP supports multiple ESCON and bus-and-tag channel interfaces.
- Higher memory capacity—The CIP offers a high memory capacity of 128 MB that can be useful for software applications, such as the TN3270 server, that have a large number of sessions.
- Port density—The CIP contains two channel interfaces in contrast to the CPA's single channel interface.

Channel Port Adapter

The CPA is available for the Cisco 7200 series routers. The CPA expands the value of Cisco's IBM channel solution by providing channel connectivity to mid-range mainframe configurations.

The CPA is a standard, single-width port adapter that provides support for IBM ESCON and bus-and-tag parallel channel attachment using the following types of interfaces:

- ESCON Channel Port Adapter (ECPA)
- Parallel Channel Port Adapter (PCPA)

Each CPA provides a single channel interface (with a single I/O connector) for Cisco 7200 series routers. In some situations, this eliminates the need for a separate FEP.

The only differences between CMCC software applications running on the CIP and a CPA are performance and capacity. The performance difference is based upon differences in the internal bus architecture of a CIP and a CPA, and the capacity difference is based on the difference in maximum memory configurations (128 MB for CIP and 32 MB for CPA). For more information about differences between the CIP and CPA, see the "Differences between the CIP and CPA" section on page 5.

The Cisco 7200 series router supports online insertion and removal (OIR), which allows you to install or remove port adapters while the system is operating.

Note In this chapter, references to CPA correspond to both the ECPA and the PCPA.

Benefits of the CPA

The CPA provides the following primary benefits:

- Cost-effective—A CPA in a Cisco 7200 series router provides industry-leading price performance.
- Simplified migration path—The CPA and CIP microcode support the same features and applications, enabling seamless migration for network expansion.
- Flexibility—The Cisco 7200 series router platform provides a great number of features and capabilities that can be used in conjunction with a CPA.

ESCON Channel Port Adapter

An ECPA is classified as a high-speed port adapter providing a single ESCON physical channel interface. Current Cisco 7200 configuration guidelines recommend using no more than three high-speed port adapters in a single Cisco 7200 router.

Refer to the *Cisco 7200 Series Port Adapter Hardware Configuration Guidelines* publication for more details.

Parallel Channel Port Adapter

A Parallel Channel Port Adapter (PCPA) provides a single parallel channel physical interface supporting 3.0 or 4.5 Mbps data transfer rates.

Differences between the CIP and CPA

Table 10 illustrates the differences between the CMCC adapters.

Table 10 Differences Between the CIP and the CPA

Product Differences	CIP	ECPA	PCPA
Router platform	Cisco 7500 Cisco 7000 with RSP7000	Cisco 7200	Cisco 7200
Channel interfaces	ESCON Parallel	ESCON	Parallel
Maximum number of interfaces	2	1	1
Maximum memory	128 MB	32 MB	32 MB
Cisco IOS release support	Cisco IOS Release 10.2 and later	Cisco IOS Release 11.3(3)T and later	Cisco IOS Release 11.3(3)T and later
Virtual port number	2	0	0
Channel interface state tracking (HSRP, SNMP alerts)	Yes	Disabled—Use the **state-tracks-signal** command to enable	Disabled—Use the **state-tracks-signal** command to enable

Supported Environments

The CMCC adapters provide support for the following environments:

- TCP/IP environments using CLAW—The Cisco IOS software implements the CLAW channel protocol to transport data between the mainframe and a CMCC adapter in TCP/IP environments.

  For more information about configuring a CMCC adapter for CLAW, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

- TCP/IP offload environments—TCP/IP offload support on a CMCC adapter provides the capability to significantly reduce the amount of overhead processing that an IBM mainframe (running the Multiple Virtual Storage (MVS), Virtual Machine (VM), or Transaction Processing Facility (TPF) operating system) must execute for handling of TCP/IP packets.

  For more information about configuring a CMCC adapter to support TCP/IP offload, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

- IP host backup environments—IP host backup support on a CMCC adapter allows the mainframe operating system to be moved from one mainframe to another without requiring a change to the router configuration at the time of the move.

  For more information about configuring a CMCC adapter for IP host backup support, see the "Configuring CLAW and TCP/IP Offload Support" chapter in this publication.

- CSNA environments—The CSNA feature on a CMCC adapter provides support for Systems Network Architecture (SNA) protocols to the IBM mainframe.

  For more information about configuring a CMCC adapter for CSNA, see the "Configuring CSNA and CMPC" chapter in this publication.

- Cisco Multipath Channel (CMPC) environments—CMPC is Cisco System's implementation of IBM's MultiPath Channel (MPC) feature on a CMCC adapter. CMPC allows VTAM to establish Advanced-Peer-to-Peer Networking (APPN) connections using both High Performance Routing (HPR) and Intermediate Session Routing (ISR) through channel-attached router platforms.

  For more information about configuring a CMCC adapter for CMPC, see the "Configuring CSNA and CMPC" chapter in this publication.

- Cisco Multipath Channel+ (CMPC+) environments—CMPC+ is Cisco System's implementation of IBM's Multipath Channel+ feature on a CMCC adapter. CMPC+ supports the MPC+ features and protocols necessary to support IP and enables High Performance Data Transfer (HPDT). It allows TCP/IP connections to the host through a CMCC adapter, using either the TCP/IP stack or the High Speed Access Services (HSAS) IP stack.

  For more information about configuring a CMCC adapter for CMPC+, see the "Configuring CMPC+" chapter in this publication.

- TN3270 server environments—The TN3270 server feature on a CMCC adapter provides a mapping between an SNA 3270 host and a TN3270 client connected to a TCP/IP network. From the perspective of an SNA 3270 host connected to the CMCC adapter, the TN3270 server is an SNA device that supports multiple PUs, with each PU supporting up to 255 logical units (LUs). From the perspective of a TN3270 client, the TN3270 server is a high-performance Telnet server that supports Telnet connections, negotiation and data format.

  For more information about configuring a CMCC adapter to support the TN3270 server, see the "Configuring the TN3270 Server" chapter in this publication.

Preparing to Configure a CMCC Adapter

This section provides guidelines to consider when preparing to configure a CMCC adapter. It includes limitations on the number of entities that you can configure on a CMCC adapter and provides information about correlating host configuration elements with your router configuration.

These guidelines are provided in the following subsections:

CMCC Configuration Guidelines

Each CMCC adapter can support the following number of configuration entities:

- A CMCC adapter can have multiple internal LANs, up to a maximum of 18.

Note Although a CMCC adapter can technically support up to 32 internal LANs, the limit of up to 18 internal adapters on a CMCC adapter makes 18 internal LANs the practical limit.

- A CMCC adapter can have multiple internal adapters, up to a maximum of 18.
- Up to 127 Service Access Points (SAPs) per internal adapter, with the Null Link Layer Service Access Point (LSAP) 0x00 reserved for the underlying MAC service access point (which is usually being used for the exchange of test frames during station discovery).

Note Several SAP values are reserved for particular protocols by the IEEE, which effectively reduces the number of SAPs available outside the router to a total of 64. This is important to remember for SAP values that you configure on the CMCC adapter for communication with network entities external to the router, so that you avoid SAP conflicts. For communication outside the router, SAP values in the range of hexadecimal 04 to 9E are recommended in increments of 4. For additional guidelines on configuring SAPs, see the "SAP Configuration Guidelines" section on page 7.

SAP Configuration Guidelines

Configuring Cisco IOS software application features on a CMCC adapter for communication with the mainframe host requires the configuration of SAPs. SAPs are used by the CMCC adapter to establish communication with the Virtual Telecommunications Access Method (VTAM) on the mainframe and to identify Logical Link Control (LLC) sessions on a CMCC's internal adapter.

To uniquely identify an LLC session, a combination of the following four entities are used in a CMCC adapter. This combination of values is sometimes referred to as the *MAC/SAP quadruple*:

- Source MAC address
- Destination MAC address
- Source SAP value
- Destination SAP value

When you are configuring SAPs on a CMCC, it is important to remember how the SAP is used in combination with these other entities to establish a unique LLC session. In order for the LLC session to be unique, there cannot be an LLC session that duplicates all four values of the MAC/SAP quadruple. In fact, only one of the values needs to be unique to qualify the particular session. Understanding this requirement is a key factor in successfully configuring a CMCC adapter to support multiple entities.

To establish the LLC sessions between external network traffic and a feature such as CSNA on a CMCC adapter in the router, an internal LAN along with an internal adapter is defined. MAC addresses are established for the internal adapters that are defined on the internal LAN in the CMCC. An internal LAN can have multiple internal adapters, and therefore, multiple MAC addresses associated with it. When LLC sessions on the CMCC are established using the same internal adapter (and therefore, the same MAC address) and are destined for the same SAP and MAC address, the source SAP must uniquely identify the session.

Consider the following guidelines when configuring SAPs on a CMCC adapter:

- If the SAP is going to be used for communication external to the router, use the following guidelines when specifying the SAP value:
  - Avoid SAPs reserved for well-known protocols.
  - Avoid a SAP of 00, which is reserved for the MAC SAP often used in the exchange of a test frame.
  - Specify SAP values in multiples of 4.

Note Some of the well-known SAP values for protocols are hexadecimal AA for SNAP, E0 for IPX, F0 for NetBIOS. For more information about some of these reserved SAP values, see Table 11 and Table 12.

- If the SAP is going to be used for communication within the router, you can maximize the number of available SAPs on an internal adapter by using multiples of 2 up to a total of 128. The CMCC adapter does not enforce the well-known values reserved for protocols and accepts any even SAP value.
- SAP 4 is commonly used as the SAP for SNA.
- CSNA can activate a maximum of 128 SAPs on the CMCC at any given time. If you are configuring the TN3270 server using a CSNA connection, the total number of SAPs open on the host plus the number of SAPs defined for PUs on the TN3270 server must be less than or equal to 128.

Reference for IEEE and Manufacturer Administered SAPs

The information in Table 11 and Table 12 is useful as a reference for understanding some of the administered LSAPs that might be encountered on the network external to the router. Remember that these are not values that the CMCC adapter enforces, and they do not specifically pertain to limitations in configuring the CMCCs.

Table 11 LSAPs Administered by IEEE

LSAP	Description
00	Null
02	Individual LLC Sublayer Management function
03	Group LLC Sublayer Management function
06	ARPANET IP
0E	Proway Network Management and Initialization
42	IEEE 802.1 Bridge Spanning-Tree Protocol
4E	EIA RS-511 Manufacturing Message Service
7E	Cisco IOS 8208 (X.25 over IEEE 802.2)
8E	Proway Active Station List
AA	Subnetwork Access Protocol (SNAP)
FE	Cisco IOS Network Layer Protocol
FF	Global LSAP

Table 12 LSAPs Implemented by Manufacturer

LSAP	Description
04	IBM SNA Path control (individual)
05	IBM SNA Path control (group)
18	Texas Instruments
80	XNS
86	Nestar
98	ARPANET (ARP)
BC	Banyan Vines
E0	Novell
F0	IBM NetBIOS
F4	IBM LAN Management (individual)
F5	IBM LAN Management (group)
F8	IBM Remote Program Load (RPL)
FA	Ungermann-Bass

Mainframe Host Configuration Considerations

Configuring a CMCC adapter and its associated features requires that you perform tasks for configuration of the mainframe and the router sides of the network environment.

Often in the mixed network environment of mainframes and LANs, an MVS systems programmer installs and maintains the mainframe side of the network, while a network engineer manages the routers on the LAN side of the network. In such an environment, the successful configuration of the CMCC adapter and its supported features requires the close coordination between these job functions at a customer site.

This section contains information for both the network engineer and the MVS systems programmer to properly configure the channel subsystem for the router and includes the following topics:

- Defining the Channel Subsystem for the Router, page 10
- Correlating Channel Configuration Parameters, page 11

Other chapters in this publication that discuss configuration of supported features on a CMCC adapter provide additional information about host-related and router-related configuration tasks associated with that feature.

Defining the Channel Subsystem for the Router

To establish the path and allocate the range of subchannel addresses that the CMCC adapter can use for communication with the mainframe, you need to specify the channel subsystem definitions in the Input/Output Control Program (IOCP) or Hardware Configuration Definition (HCD) on the host.

The following sample configuration shows the CHPID, CNTLUNIT, and IODEVICE statements that might be defined in an IOCP file for parallel channels and ESCON channels on the CIP or CPA. The parameters in bold indicate values that might vary by the type of channel being defined.

```
**************************************************************************************
* Parallel channel--CIP or CPA may be subchannel addresses 580-58F
**************************************************************************************
CHPID    PATH=((21)),TYPE=BL
CNTLUNIT CUNUMBER=0580,PATH=(21),UNIT=3088,UNITADD=((80,16)),SHARED=N,PROTOCOL=S4
IODEVICE ADDRESS=(580,16),CUNUMBER=(0580),UNIT=CTC
**************************************************************************************
* ESCON channel--CIP or CPA may be subchannel addresses D00-D0F
**************************************************************************************
CHPID    PATH=((1F)),TYPE=CNC
CNTLUNIT CUNUMBER=0D00,PATH=(1F),UNIT=3172,UNITADD=((00,16))
IODEVICE ADDRESS=(D00,16),CUNUMBER=(0D00),UNIT=SCTC
**************************************************************************************
* ESCON channel with ESCON director--CIP or CPA may be subchannel addresses 700-70F
**************************************************************************************
CHPID    PATH=((1C)),TYPE=CNC,SWITCH=01
CNTLUNIT CUNUMBER=0700,PATH=(1C),UNIT=3172,UNITADD=((00,16)),LINK=(C4)
IODEVICE ADDRESS=(700,16),CUNUMBER=(0700),UNIT=SCTC
```

The subchannel parameters differ by the type of channel that you are defining. For example, to support a CMCC parallel channel always use the channel type BL for block multiplexor, data streaming mode. ESCON channels use a channel type of CNC for Native ESCON (or type CVC might be used if an ESCON Converter is in use).

In addition, the UNIT types specified in the CNTLUNIT and IODEVICE statements differ for parallel and ESCON channels. The ESCON director also implements the additional parameters for SWITCH and LINK to identify a number for the ESCON director and specify the port in the ESCON director to which the router is connected.

Note In the format of a real IOCP file, all of the CHPID, CNTLUNIT, and IODEVICE statements are organized into separate groups, and are not listed one after the other as shown. You can correlate the statements in a real IOCP file by using the PATH parameter to associate the CHPID definition with a corresponding CNTLUNIT statement, and using the CUNUMBER parameter to correlate the IODEVICE and the CNTLUNIT statements.

Correlating Channel Configuration Parameters

This section provides detailed information about correlating values found in the VM and MVS system I/O configuration files with the arguments required in the **claw**, **csna**, **cmpc**, and **offload** interface configuration commands on the CMCC adapter.

To properly configure the channel subsystem on the router side you need to know the following information:

- Channel path, including any of the following values when applicable:
  - ESCON director output port to the mainframe
  - LPAR number
  - CUADD value
- Unit address

This information is defined on the host in the IOCP. In versions of MVS 5.2 and later, an HCD might be used as an alternative method to define this information. To locate this information or to configure it on the mainframe host, contact your site's systems programmer.

Determining the Path Argument

When you define CLAW, CSNA, CMPC or CMPC+, and Offload parameters on a CMCC adapter, you must supply path information and device address information to support routing on an IBM channel. The path information can be simple, in the case of a channel directly attached to a router using bus and tag cables, or more complex when the path includes an ESCON director switch or multiple image facility (EMIF) support.

This example shows the syntax for the CMCC adapter commands that require subchannel information, which is configured in the *path* and *device* arguments of the following commands:

```
claw path device ip-address host-name device-name host-app device-app [broadcast]

csna path device [maxpiu value][time-delay value][length-delay value]

cmpc path device tg-name {read | write}

offload path device ip-address host-name device-name host-ip-link device-ip-link
host-api-link device-api-link [broadcast] [backup]
```

The *path* argument in each of the commands is a four-digit hexadecimal value that concatenates the path value (2 digits), EMIF partition number (1 digit), and control unit logical address (1 digit) as described in Table 13.

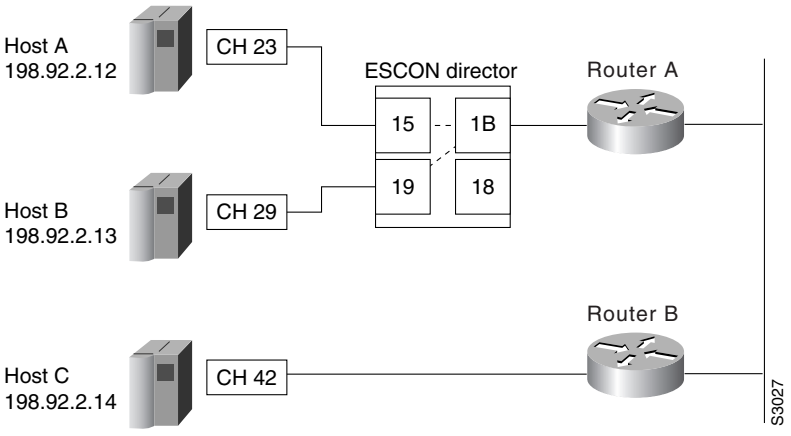
For bus and tag channel connections, the *path* value is always 0100. You do not need the information in Table 13 to determine the *path* value.

Table 13 Breakdown of Path Argument Values

Path Argument Breakdown	Values	Description
Path digits	01–FF	For a directly attached ESCON channel or any Parallel channel this value is 01, *unless* a systems programmer has configured another value. For a channel attached through an ESCON director switch, specify the outbound port number in the first two digits of the *path* argument. This is the port which, from the router point of view, exits the switch and attaches to the host.
EMIF partition number digit	0–F	For a Parallel channel, this value is 0. For a directly attached ESCON channel, the value might be non-zero. If the host is running in Logical Partition (LPAR) mode and the CHPID is defined as shared, specify the partition number in this digit of the *path* argument.
Control unit logical address digit	0–F	For a Parallel channel, this value is 0. For a directly attached ESCON channel, the value might be non-zero. If the CUADD value is specified in the IOCP CNTLUNIT statement, specify that value in this digit of the *path* argument.

Consider the network configuration in Figure 247, where two host systems connect to the ESCON director switch on paths 23 and 29. The channels both exit the switch on path 1B and attach to Router A. Note that the path between Host A and Host B is dynamically switched within the ESCON director. Host C is attached directly to Router B through path 42.

Figure 247 System with an ESCON Director Switch and a Directly Attached Channel



The IOCP control unit statements to configure the channel paths shown in Figure 247 might look similar to the following sample configuration statements:

Sample IOCP Control Unit Statements for Host A

```
CNTLUNIT CUNUMBER=0001, PATH=(23), LINK=1B, UNITADD=((00,64)), UNIT=SCTC, CUADD=F
```

Sample IOCP Control Unit Statements for Host B

```
CNTLUNIT CUNUMBER=0002, PATH=(29), LINK=1B, UNITADD=((00,64)), UNIT=SCTC, CUADD=A
```

Sample IOCP Control Unit Statements for Host C

```
CNTLUNIT CUNUMBER=000A, PATH=(42), UNIT=SCTC, UNITADD=((00,64))
```

Note A mainframe systems programmer can provide you with the actual IOCP values for your site's configuration.

Using the above IOCP values as an example and following the guidelines provided in Table 13, the following *path* argument is used as shown in the example **csna** or **cmpc** commands for the two channel attachments to Router A:

```
csna 150F
csna 190A

cmpc 150F
cmpc 190A
```

In Figure 247 the ESCON director ports 15 and 19 are the channel attachments from the ESCON director to each host. Note that the outbound ports from the ESCON director to the host are the values used in the first 2 digits of the *path* argument.

The following *path* argument is used for the directly attached channel to Router B, as shown in the example **csna** or **cmpc** commands:

```
csna 0100

cmpc 0100
```

Determining the Device Argument

When you define CLAW, CSNA, CMPC or CMPC+, and Offload parameters on a CMCC adapter, you must supply path information and device address information to support routing on an IBM channel. To determine the value for the *device* argument in the **claw**, **csna**, **cmpc**, or **offload** interface configuration commands on the CMCC adapter, find the UNITADD parameter in the host IOCP definition.

The UNITADD parameter in the CNTLUNIT macro of the IOCP file defines the valid range for device addresses. For example, a UNITADD parameter of (00,64) means that the first valid device address is 00 and the number of devices is 64. In the hexadecimal notation used by channel configuration commands this translates to a range of 00 to 3F.

Using that unit address information, the example **csna** and **cmpc** commands now add values for the *device* arguments to the two channel attachments to Router A:

```
csna 150F 00
csna 190A 01

cmpc 150F 02
cmpc 150F 03

cmpc 190A 03
```

```
cmpc 190A 04
```

The following example **csna** and **cmpc** commands show the *path* and *device* arguments for the directly attached channel to Router B:

```
csna 0100 00

cmpc 0100 01
cmpc 0100 02
```

Note In this example, if you configure CSNA and CMPC on the same CMCC port then you must use unique unit addresses. Also, CMPC requires two unit addresses for the *device* argument. One unit address is used in a **cmpc** command to define the read subchannel, and one is used in a second **cmpc** command to define the write subchannel. The device addresses do not need to be consecutive.

Determining the Device Argument from an IODEVICE Address

When you have a directly attached channel, the mainframe systems programmer might provide you with a system IODEVICE ADDRESS that you can use to determine the required subchannel information. In this case, you must work backwards through the IOCP file to locate the proper *device* argument value for the CMCC adapter interface commands.

Example 1

In this first example, the IODEVICE ADDRESS value is 800. Using this number you can locate the IODEVICE ADDRESS statement in the IOCP file, which points you to the CNTLUNIT statement that contains the *device* argument values for the **claw**, **csna**, **cmpc** or **offload** commands:

```
IODEVICE ADDRESS=(0800,256),CUNUMBR=(0012),UNIT=SCTC
**** Address 800 points to CUNUMBR 0012 in the following statement

CNTLUNIT CUNUMBR=0012,PATH=(28),UNIT=SCTC,UNITADD=((00,256))
**** A valid value for the device argument is the UNITADD value of 00
```

From this example, the **csna** command would be similar to the following:

```
csna 0100 00
```

Example 2

In this example the mainframe systems programmer provides an available IODEVICE ADDRESS of 350, which does not directly correspond to a value in the IOCP file, but is within a range of 64 addresses beginning at device address 340 (as shown in the IODEVICE ADDRESS=(340,64) statement). The value 350 is at an offset of 10 from the beginning value of 340 in this statement:

```
IODEVICE ADDRESS=(0340,64),CUNUMBR=(0008),UNIT=SCTC
IODEVICE ADDRESS=(0380,64),CUNUMBR=(0009),UNIT=SCTC
**** Address 350 is in the range of 64 addresses beginning at address 340 corresponding
**** to CUNUMBER 0008

CNTLUNIT CUNUMBR=0008,PATH=(24),UNIT=SCTC,UNITADD=((40,64)),SHARED=N, X
**** The device is the UNITADD value of 40, offset by 10, which is 50
```

To determine the unit address for the *device* argument value in the **claw**, **csna**, **cmpc** or **offload** commands, you must use the same offset that you determined for the IODEVICE ADDRESS and calculate the UNITADD parameter from the corresponding CNTLUNIT statement. In this example, CUNUMBR=0008 is the corresponding CNTLUNIT statement for IODEVICE ADDRESS 350. The first

unit address in that CNTLUNIT statement is 40 (in parameter UNITADD), which correlates to the first IODEVICE ADDRESS of 340. To determine the corresponding unit address for IODEVICE ADDRESS 350, determine the value at offset 10 from 40, which is 50.

In this example, the **csna** command would be similar to the following:

```
csna 0100 50
```

Note In the IOCP examples for the IODEVICE and CNTLUNIT statements, UNIT=SCTC is the usual value for ESCON channels. Parallel channels will have UNIT=3088 in the CNTLUNIT statement and UNIT=CTC in the IODEVICE statement.

Tip You can prevent configuration problems and more readily correlate configuration between the router and the host if you follow the convention of using the last two digits of the starting IODEVICE ADDRESS as the starting value for the range of unit addresses in the UNITADD parameter of the CNTLUNIT statement. For example, if you use IODEVICE ADDRESS=(410,8) then use "10" as the beginning value of the unit address as in UNITADD=((10,8)). To avoid confusion and potential configuration errors, do not specify IODEVICE ADDRESS=(410,8) and then begin the unit addresses at value 00.

Disabling the Missing Interrupt Handler

Because the appropriate configuration of the missing interrupt handler (MIH) varies according to the protocols and software releases used, Cisco offers the following guidance:

- For OS/390 releases Version 2 Release 4 and earlier, set the MIH to zero.
- For OS/390 releases later than Version 2 Release 4 and z/OS releases, refer to the following section of the z/OS Communications Server IP Configuration Reference: http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/f1a1b420/1.2.13?SHELF=f1a1bk31&DT=20020604120755#HDRMOLLY

This section includes the following topics:

- Disabling the MIH on Mainframes Running MVS, page 15
- Disabling the MIH on Mainframes Running VM, page 16

For additional information about disabling the MIH, refer to the IBM publication *Transmission Control Protocol/Internet Protocol TCP/IP Version 2 Release 2.1 for MVS: Planning and Customization* (publication SC31-6085 or later).

Disabling the MIH on Mainframes Running MVS

To disable the MIH on an MVS host, you need to configure a statement in the IECIOS*xx* member of the SYS1.PARMLIB partitioned dataset. To properly identify the IECIOS*xx* member to use, there must be a corresponding statement IOS=*xx* in the member IEASYS00 (where 00 is the default suffix).

Note The statement IOS=*xx* specifies that *xx* is the suffix of the IECIOS member that contains the configuration. For example, the statement IOS=01 points to the IECIOS01 member. If this statement is not included in the IEASYS file, you can specify it dynamically using the **/SET IOS=***xx* command on the command line. For more information, see your site's systems programmer.

To disable the MIH on an MVS host, perform the following steps:

Step 1 Type the following statement in the IECIOS*xx* member of SYS1.PARMLIB, where *yyy-yyy* specifies the range of unit addresses for which you want to disable the MIH:

```
MIH TIME=00:00:00, DEV=(yyy-yyy)
```

This configures the MVS host to disable the MIH every time that MVS is restarted (an Initial Program Load (IPL) is performed).

Step 2 To dynamically change the MIH value for the currently active MVS operating system, issue the following command at the command line:

```
/SETIOS MIH DEV=xxx, TIME=00:00
```

Step 3 To display the currently enabled MIH value, issue the following command at the command line:

```
/D IOS,MIH
```

Note If you are using Dynamic Reconfiguration Management (DRM), type DYNAMIC=NO in the device statement. This value can be YES for IBM TCP/IP version 3.2.

Disabling the MIH on Mainframes Running VM

To disable the MIH on a VM host, you need to configure a statement in the PROFILE EXEC for the AUTOLOG1 userid (or equivalent userid for your site).

To disable the MIH on a VM host, perform the following steps:

Step 1 Type the following statement in the PROFILE EXEC of the AUTOLOG1 (or equivalent) userid, where *yyy-yyy* specifies the range of unit addresses for which you want to disable the MIH:

```
SET MITIME yyy-yyy 00:00
```

or

```
SET MITIME yyy-yyy OFF
```

This configures the VM host to disable the MIH every time that VM is restarted (an Initial Program Load (IPL) is performed).

Step 2 To dynamically change the MIH value for the currently active VM operating system, issue either of the commands shown in Step 1 at the command line.

Step 3 To display the currently enabled MIH value, issue the following command at the command line:

```
Q MITIME
```

Note For VM or MVS Guests under VM, code V=R (Real mode) for the Guest so that the CLAW channel programs build properly. For more information, see your site's systems programmer.

CMCC Adapter Configuration Task List

This section describes some of the global tasks that apply to configuring any CMCC adapter. Information about configuring features on a CMCC adapter are described in the related chapters of this guide.

This section includes the following configuration tasks:

- Loading the CMCC Adapter Microcode Image, page 17
- Selecting the Interface, page 22
- Selecting a Data Rate for the Parallel Channel Interfaces, page 23
- Configuring Channel Interface Tracking for HSRP or SNMP Alerts, page 24

See the "CPA Microcode Load Configuration Examples" section on page 30 for examples.

Loading the CMCC Adapter Microcode Image

This section provides information on loading, upgrading and verifying the microcode images for the CIP and CPA in the following topics:

- Loading the CIP Microcode Image for All Adapters in the Router, page 17
- Upgrading the CIP Microcode Image, page 19
- Upgrading the CPA Microcode Image for All Adapters in the Router, page 20
- Upgrading the CPA Microcode Image for a Particular Adapter, page 21
- Verifying the CIP and CPA Microcode Image, page 21

Loading the CIP Microcode Image for All Adapters in the Router

Beginning with Cisco IOS Release 11.1, the CIP microcode (or CIP *image*) no longer is bundled with the Cisco IOS software. You must have Flash memory installed on the Route Switch Processor (RSP) card to use the IBM channel-attachment features in Cisco IOS Release 11.1 and later.

The CIP image is preloaded on Flash cards for all Cisco 7000 with RSP7000 and Cisco 7500 series routers ordered with the CIP option for Cisco IOS Release 11.1 and later.

Use the commands in this section if you are loading the CIP microcode image for the first time, or for all adapters in your router.

Caution Using the **microcode reload** command as shown in step 5 forces a microcode reload on all adapters in the router and shuts down the router. Do not use this command if you are on a production network and are not prepared for a router outage.

To prepare the CIP, use the following commands beginning in privileged EXEC command mode:

	Command	Purpose
Step 1	`Router> enable`	Enters the privileged EXEC mode command interpreter.
Step 2	`Router# copy tftp:`*filename* `[bootflash: | slot0: | slot1:]`*filename*	Copies the CIP microcode image from a server to either of the Flash memory cards. The source of the file is **tftp**:*filename*. Use the appropriate command for your system. You must be running Cisco IOS Release 11.1 or later prior to executing a **copy tftp** command.
Step 3	`Router# configure terminal`	From privileged EXEC command mode, enters global configuration mode and specifies that the console terminal is the source of the configuration subcommands.
Step 4	`Router(config)# microcode cip flash slot`*n:cipxx-yy* or `Router(config)# microcode cip flash bootflash:`*cipxx-yy*	Configures the router to load the Flash image to the CIP: • Enters global configuration mode and specifies that the CIP microcode loads from a Flash card in router slot *n* or from embedded Flash. • Loads the image from Flash to the CIP card.
Step 5	`Router(config)# microcode reload`	Forces a microcode reload for all adapters in the router. **Note** This command shuts down the router if you are on a live network.
Step 6	`Router(config)# end`	Exits global configuration mode.
Step 7	`Router# copy running-config startup-config`	Saves the running configuration as the new startup configuration in NVRAM.

Upgrading the CIP Microcode Image

Beginning with Cisco IOS Release 11.1, the CIP microcode (or CIP *image*) no longer is bundled with the Cisco IOS software. You must have Flash memory installed on the RSP card to use the IBM channel-attachment features in Cisco IOS Release 11.1 and later.

The CIP image is preloaded on Flash cards for all Cisco 7000 with RSP7000 and Cisco 7500 series routers ordered with the CIP option for Cisco IOS Release 11.1 and later.

Use the commands in this section if you are upgrading the CIP image in your router.

To upgrade the CIP microcode, use the following commands beginning in privileged EXEC command mode:

	Command	Purpose
Step 1	`Router> enable`	Enters the privileged EXEC mode command interpreter.
Step 2	`Router# copy tftp:`*`filename`* `[bootflash: | slot0: | slot1:]`*`filename`*	Copies the CIP microcode image from a server to either of the Flash memory cards. The source of the file is **tftp**:*filename*. Use the appropriate command for your system. You must be running Cisco IOS Release 11.1 or later prior to executing a **copy tftp** command.
Step 3	`Router# configure terminal`	From privileged EXEC command mode, enters global configuration mode and specifies that the console terminal is the source of the configuration subcommands.
Step 4	`Router(config)# microcode cip flash slot`*`n:cipxx-yy`* or `Router(config)# microcode cip flash bootflash:`*`cipxx-yy`*	Configures the router to load the Flash image to the CIP: • Enters global configuration mode and specifies that the CIP microcode loads from a Flash card in router slot *n* or from embedded Flash. • Loads the image from Flash to the CIP card.
Step 5	`Router(config)# end`	Exits global configuration mode.
Step 6	`Router# copy running-config startup-config`	Saves the running configuration as the new startup configuration in NVRAM.

Upgrading the CPA Microcode Image for All Adapters in the Router

The CPA microcode image is preloaded on Flash memory cards for Cisco 7200 series routers for Cisco IOS Release 11.3(3)T and later. You may be required to copy a new image to Flash memory when a new microcode image becomes available. Use the commands in this section if you are upgrading or loading a microcode image other than the default image for all adapters in the router.

Caution Using the **microcode reload** command as shown in Step 5 forces a microcode reload on all interfaces in the router and shuts down the router. Do not use this command if you are on a production network and are not prepared for a router outage.

To prepare the CPA, use the following commands beginning in privileged EXEC command mode:

	Command	Purpose
Step 1	`Router> enable`	Enters the privileged EXEC mode command interpreter.
Step 2	`Router# copy tftp:`*filename* `[bootflash: | slot0: | slot1:]`*filename*	Copies the CPA microcode image from a server to either of the Flash memory cards. The source of the file is **tftp**:*filename*. Use the appropriate command for your system. You must be running Cisco IOS Release 11.1 or later prior to executing a **copy tftp** command.
Step 3	`Router# configure terminal`	From privileged EXEC command mode, enters global configuration mode and specifies that the console terminal is the source of the configuration subcommands.
Step 4	`Router(config)# microcode {ecpa | pcpa} slot`*n:xcpaxx-yy*	Loads the microcode from an individual microcode image that is stored as a file on a Flash memory card on a CPA adapter. The **slot** argument of the command specifies the slot location and filename of the microcode image, such as slot0:xcpa26-1.

Step 5	`Router(config)# microcode reload`
	or
	`Router# microcode reload all`
		From global configuration mode, loads the CPA microcode image for all of the adapters in the router.
		or
		From privileged EXEC mode, forces a microcode reload for all CPA adapters.
		Note These commands shut down the router if you are on a live network.

Step 6	`Router(config)# end`	Exits global configuration mode.
Step 7	`Router# copy running-config startup-config`	Saves the running configuration as the new startup configuration in NVRAM.

Upgrading the CPA Microcode Image for a Particular Adapter

The CPA microcode image is preloaded on Flash memory cards for Cisco 7200 series routers for Cisco IOS Release 11.3(3)T and later. You may be required to copy a new image to Flash memory when a new microcode image becomes available. Use the commands in this section if you are upgrading or loading a microcode image other than the default image for a particular CPA adapter.

To prepare the CPA, use the following commands beginning in privileged EXEC command mode:

	Command	Purpose
Step 1	`Router> enable`	Enters the privileged EXEC mode command interpreter.
Step 2	`Router# copy tftp:`*filename* `[bootflash: | slot0: | slot1:]`*filename*	Copies the CPA microcode image from a server to either of the Flash memory cards. The source of the file is **tftp**:*filename*. Use the appropriate command for your system. You must be running Cisco IOS Release 11.1 or later prior to executing a **copy tftp** command.
Step 3	`Router# configure terminal`	From privileged EXEC command mode, enters global configuration mode and specifies that the console terminal is the source of the configuration subcommands.
Step 4	`Router(config)# microcode {ecpa | pcpa} slot`*n:xcpaxx-yy*	Loads the microcode from an individual microcode image that is stored as a file on a Flash memory card on a CPA adapter. The **slot** argument of the command specifies the slot location and filename of the microcode image, such as slot0:xcpa26-1.
Step 5	`Router# microcode reload {all | {{ecpa | pcpa} [slot` *number*`]}`	From privileged EXEC mode, forces a microcode reload for a specific interface in a particular slot in a CPA adapter. This command allows you to reset a particular card without resetting every card in the router. **Note** The **all** keyword reloads all adapters in the router.
Step 6	`Router(config)# end`	Exits global configuration mode.
Step 7	`Router# copy running-config startup-config`	Saves the running configuration as the new startup configuration in NVRAM.

Verifying the CIP and CPA Microcode Image

When a router is starting and the bootflash is loading, the router searches for the default CIP or CPA microcode associated with the current bootflash image. This microcode image might not be the current image that you have loaded. This produces some messages that seem to indicate that the microcode is not loading properly. However, these messages (as shown in the following example for the CIP) are part of the normal loading process:

```
*Oct  1 13:37:28.078: %SYS-5-RELOAD: Reload requested
System Bootstrap, Version 11.1(2) [nitin 2], RELEASE SOFTWARE (fc1)
```

```
Copyright (c) 1994 by cisco Systems, Inc.
SLOT 2 RSP2 is system master
RSP2 processor with 65536 KB of main memory

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCreading the file into memory...
Self decompressing the image :
##########################################################################################
##########################################################################################
################### [OK]

%DBUS-3-NOSUPPORT: No driver support in this image for CIP2 in slot 4 (card type 13) -
disabled
%RSP-3-NOSTART: No microcode for CIP2 card, slot 4
%SYS-4-CONFIG_NEWER: Configurations from version 12.0 may not be correctly understood.
%DBUS-3-NOSUPPORT: No driver support in this image for CIP2 in slot 4 (card type 13) -
disabled
%RSP-3-NOSTART: No microcode for CIP2 card, slot 4
```

The Cisco IOS software loads either the default CIP or CPA microcode image, or the image specified in the **microcode** command. If you want to confirm the microcode image that is loaded on the router, use the **show controllers cbus** command for the CIP or the **show controllers channel** command for the CPA to display the loaded microcode images.

To verify the CIP or CPA microcode, use the following commands beginning in privileged EXEC command mode:

	Command	Purpose
Step 1	`Router# show controllers cbus`	Displays images loaded on the CIP card.
Step 2	`Router# show controllers channel` *slot/port*	Displays the currently loaded and running microcode version for each CPA in the router.
Step 3	`Router# show running-config`	Displays the contents of the configuration file.

Note Do not use the **show microcode** command to verify the loaded microcode image. The **show microcode** command displays the default microcode images for the version of Cisco IOS software that you are running, which does not necessarily correspond to the image that you have loaded.

Selecting the Interface

Some CMCC adapter software features are configured on a *virtual port*. On the CIP adapter cards installed in a Cisco 7000 with RSP7000 or a Cisco 7500 series router, there are up to two physical ports, numbered 0 and 1, and a virtual port, numbered 2. However, on the CPA installed in a Cisco 7200 series router, the single physical port and the virtual port are configured using the same port number ID, number 0.

Before you configure a channel-attached interface, you must select the interface. Use the following command in global configuration mode:

Command	Purpose
`Router(config)# interface channel slot/port`	Selects the interface and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*Port* value corresponds to 0 or 1 for the physical interface, and 2 for the virtual interface. • CPA—*Port* value corresponds to port 0.

Use the **show extended channel** EXEC commands to display current CMCC adapter status. This command provides a report for each interface configured to support IBM channel attachment.

Selecting a Data Rate for the Parallel Channel Interfaces

When you configure a parallel channel-attached interface (such as a PCA on a CIP or a PCPA on a CPA) that uses bus-and-tag connections, you can specify a data rate of either 3 MBps or 4.5 MBps.

Note that the unit of measure for this command is mega*bytes* per second (MBps). When you use the **show interface channel** command, the data rate is shown in the BW field (for bandwidth) in kilo*bits* per second (kbps). For example, a channel data rate of 3 MBps is shown as 36864 kbps in the output for the **show interface channel** command.

To configure the parallel data rate on the router, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# channel-protocol [s | s4]`	(Optional) Defines the data transfer rate for parallel channel interfaces. The available options for this command are: • **s**—Specifies that the parallel channel interface operates at a rate of 3 MBps, or 36864 kbps. This is the default. • **s4**—Specifies that the parallel channel interface operates at a rate of 4.5 MBps.

Mainframe Configuration Tip

The **channel-protocol** command has a corollary parameter called PROTOCOL in the CNTLUNIT statement of the mainframe IOCP definition. The PROTOCOL parameter in the IOCP definition specifies the maximum speed of a bus-and-tag channel connection for the corresponding CSNA device in the router. Note that even if the CNTLUNIT statement in the IOCP specifies a value of PROTOCOL=S4 (4.5 MBps), the channel interface will operate at 3 MBps if at the router you use the default value or specify **s** in the **channel-protocol** command. Therefore, if you want to configure a channel speed of 4.5 MBps, be sure to specify a value of **s4** for both the PROTOCOL parameter in the IOCP and the **channel-protocol** command in the router.

Configuring Channel Interface Tracking for HSRP or SNMP Alerts

If you want to use Hot Standby Router Protocol (HSRP) or SNMP alerts to monitor channel interface status for an ECPA or PCPA channel interface, use the following command in interface configuration mode to enable physical interface signal tracking:

Command	Purpose
`Router(config-if)# state-tracks-signal`	Enables tracking of the physical interface signal for an ECPA or PCPA channel interface.

The **state-tracks-signal** command is valid only on channel interfaces which combine the functions of both a physical and virtual interface. The ECPA and PCPA are examples of this type of channel interface. The command is not valid for the CIP, which has a separate channel interface for the virtual channel functions.

Monitoring and Maintaining a CMCC Adapter

You can perform the tasks in the following sections to monitor and maintain the interfaces:

Monitoring Interface Status

To display information about the interface, including the version of the software and the hardware, the controller status, and statistics about the interfaces, you can use the show commands listed in the following table. To see the full list of **show** commands supported, enter **show ?** at the EXEC prompt.

Perform the following commands in EXEC mode to display information associated with each command. All commands are applicable to all CMCC adapter interfaces (CIP and CPA), unless it is mentioned that they are specific to a particular CMCC adapter. Commands are listed in alphabetic order.

Command	Purpose
Router# **show controllers cbus**	Displays the cbus internal state for the Cisco 7000 with RSP7000 and Cisco 7500 series routers. Also included in the display is CIP-specific information such as the currently loaded microcode, currently loaded microcode application segments, and load metrics (such as CPU and memory statistics).
Router# **show controllers channel** [*slot/port*]	Displays CPA-specific information, including the currently loaded microcode.
Router# **show extended channel** *slot/port* **backup** [*ip-address*]	Displays information about CLAW and offload commands for each backup group.
Router# **show extended channel** *slot/port* **cmgr** [*tg-name*]	Displays information about the MPC+ TG connection manager.
Router# **show extended channel** *slot/port* **cmpc** [*path* [*device*]]	Displays information about each CMPC or CMPC+ subchannel configured on the specified CMCC adapter interface.
Router# **show extended channel** slot/*port* **connection-map llc2**	Displays the number of active LLC2 connections for each SAP and the mapping of the internal MAC adapter and the SAP to the resource that activated the SAP.
Router# **show extended channel** *slot/port* **csna** [**admin** | **oper** | **stats**] [*path* [*device*]]	Displays information about the CSNA subchannels configured on the specified CMCC adapter interface.
Router# **show extended channel** *slot/port* **llc2** [**admin** | **oper** | **stats**] [*lmac* [*lsap* [*rmac* [*rsap*]]]]	Displays information about the LLC2 sessions running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **max-llc2-sessions**	Displays information about the number of LLC2 sessions supported on the CMCC adapter.
Router# **show extended channel** *slot/port* **icmp-stack** [*ip-address*]	Displays information about the ICMP stack running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **ip-stack** [*ip-address*]	Displays information about the IP stack running on the CMCC adapter interfaces
Router# **show extended channel** *slot/port* **llc2** [**admin** | **oper** | **stats**] [*lmac* [*lsap* [*rmac* [*rsap*]]]]	Displays information about the LCC2 sessions running on the CMCC adapter interfaces.
Router# **show extended channel** slot/*port* **packing names** [*path* [*device*]]	Displays CLAW packing names and their connection state.
Router# **show extended channel** *slot/port* **packing stats** [*path*[*device*]]	Displays CLAW packing statistics.
Router# **show extended channel** *slot/port* **statistics** [*path* [*device*]] [**connected**]	Displays information about CMCC adapter interfaces for diagnostic purposes.
Router# **show extended channel** *slot/port* **subchannel** [**connected**]	Displays information about the CMCC adapter interfaces

Command	Purpose
Router# **show extended channel** *slot/port* **tcp-connections** [[*loc-ip-addr* [*loc-port* [*rem-ip-addr* [*rem-port*]] [**detail** | **summary**]	Displays information about the TCP sockets on a channel interface.
Router# **show extended channel** slot/*port* **tcp-stack** [*ip-address*]	Displays information about the TCP stack running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **tg** [**oper** | **stats**] [**detailed**] [*tg-name*]	Displays configuration, operational, and statistics information for CMPC and CMPC+ transmission groups configured on a specified CMCC adapter internal LAN interface.
Router# **show extended channel** *slot/port* **tn3270-server**	Displays current configuration parameters and the status of the PUs defined in each TN3270 server.
Router# **show extended channel** *slot/port* **tn3270-server client-ip-address** *ip-address* [**disconnected** | **in-session** | **pending**]	Displays information about all clients at a specific IP address.
Router# **show extended channel** *slot/port* **tn3270-server dlur**	Displays information about the SNA session switch.
Router# **show extended channel** *slot/port* **tn3270-server dlurlink** *name*	Displays information about the DLUR components.
Router# **show extended channel** *slot/port* **tn3270-server nailed-ip** *ip-address*	Displays mappings between a nailed client IP address and nailed LUs.
Router# **show extended channel** *slot/virtual channel* **tn3270-server pu** *pu-name* [**cluster**]	Displays information about the client LUs associated with a specified PU including the cluster layout and pool name.
Router# **show extended channel** *slot/port* **tn3270-server pu** *pu-name* **lu** locaddr [**history**]	Displays information about the TN3270 server LUs running on CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **udp-listeners** [*ip-address*]	Displays information about the UDP listener sockets on the CMCC adapter interfaces.
Router# **show extended channel** *slot/virtual channel* **tn3270-server response-time application** [*appl-name* [**detail**]]	Displays information about each client group application for the specified VTAM appl name. List each member of the client group with its individual response-time statistics.
Router# **show extended channel** *slot/virtual channel* t**n3270-server response-time global**	Displays information about the global client groups.
Router# **show extended channel** *slot/virtual/channel* **tn3270-server response-time link** [*link-name*]	Displays information about the specified per-host-link client group.
Router# **show extended channel** *slot/virtual* channel **tn3270-server response-time listen-point**	Displays information about listen-point type client groups.
Router# **show extended channel** *slot/virtual* channel **tn3270-server response-time subnet** [*ip-mask* [**detail**]]	Displays information about the specified client group.
Router# **show extended channel** *slot/port* **udp-stack** [*ip-address*]	Displays information about the UDP stack running on the CMCC adapter interfaces.

Command	Purpose
`Router# show interfaces channel slot/port accounting`	Displays the number of packets for each protocol type that has been sent through the channel interface.
`Router# show version`	Displays the hardware configuration, software version, names and sources of configuration files, and boot images.

Clearing and Resetting an Interface

There are several commands that you can use on a CMCC adapter to clear statistics counters by interface or by feature, or to reset the hardware logic on an interface.

Clearing Interface Statistics Counters

To clear the statistics counters that are displayed in the output of the **show interfaces** command, use the following command in EXEC mode:

Command	Purpose
`Router# clear counters [type slot/port]`	Clears interface counters on the router.

Note This command does not clear counters retrieved using Simple Network Management Protocol (SNMP), but only those seen with the EXEC **show interfaces** command.

Clearing Feature-Specific Statistics Counters

You can reset the statistics counters that are displayed in the output of the **show extended channel** commands by a particular feature on the interface.

To clear the counters associated with application features configured on the CMCC adapters, use the following command in EXEC mode:

Command	Purpose
`Router# clear extended counters channel slot/port [csna | icmp-stack | ip-stack | llc2 | statistics | tcp-connections | tcp-stack | tg | tn3270-server | udp-stack]`	Clears counters for application features configured on CMCC adapters.

Note This command does not clear counters retrieved using Simple Network Management Protocol (SNMP), but only those seen with the EXEC **show extended channel** commands.

Clearing the Hardware Logic on an Interface

Under normal circumstances, you do not need to clear the hardware logic on interfaces. However, if it is necessary to clear the hardware logic on an interface, use the following command in EXEC mode:

Command	Purpose
`Router# clear interface [type slot/port]`	Resets the hardware logic on an interface.

Monitoring the Physical Channel Interface on the CPA

Unlike the CIP, which has a separate channel interface for the virtual channel functions, the ECPA and PCPA have a single interface that combines the functions of both a physical and virtual channel interface. For this reason, monitoring the physical channel interface on a CPA requires other considerations in its implementation.

In Cisco IOS releases prior to 12.0(4.1), you could not configure how the state of the physical interface on a CPA was tracked, particularly when the interface was configured for **no shutdown**. In those previous Cisco IOS releases when the CPA channel interface was configured for **no shutdown**, the channel interface status was always reported as UP/UP, even when no signal was present on the physical connection.

In Cisco IOS Release 12.0(4.1) and later, you can use the **state-tracks-signal** configuration command to control how you want the state of the CPA's channel interface to be reported. The **state-tracks-signal** command is useful in environments where you are using HSRP or SNMP alerts to monitor channel interface status.

To enable physical interface signal tracking, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# state-tracks-signal`	Enables tracking of the physical interface signal for an ECPA or PCPA channel interface.

When the **state-tracks-signal** command is used on an interface that is configured for **no shutdown**, then the state of the channel interface is reported according to the status of the physical channel interface signal. If the physical channel interface signal is not present, then the channel interface status is DOWN/DOWN.

When the channel interface is configured for **no state-tracks-signal** (the default) and **no shutdown**, the channel interface status is always reported as UP/UP, even when there is no signal present on the physical connection. This configuration is useful for TN3270 server environments that are operating in a mode without any physical channel interface connections.

Shutting Down and Restarting an Interface

You can disable an interface on a CMCC adapter. Disabling an interface disables all of the functions on the specified interface and marks the interface as unavailable on all monitoring command displays. This information is communicated to other network servers through all dynamic routing protocols. The interface will not be mentioned in any routing updates. On a CMCC adapter with an ESCON interface,

a command is sent to the host to inform it of the impending shutdown. On the CMCC adapter's Parallel interface, the **shutdown** command disables the adapter card's transceivers and the interface stops responding to all commands.

It is recommended that you shut down a channel interface for some of the following reasons:

- For a CMCC adapter's ESCON interface, to change the interface type of a Cisco 7000 with RSP7000 or Cisco 7500 port online. To ensure that the system recognizes the new interface type, shut down the interface and then reenable it after changing the interface. Refer to your hardware documentation for more details.
- If you want to reload the router
- If, prior to reloading the microcode, you want to shut down the interface
- If you want to power off the router
- If it is recommended that a channel interface be shut down

To shut down an interface and then restart it, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# shutdown`	Shuts down an interface.
Step 2	`Router(config-if)# no shutdown`	Enables an interface.

To check whether an interface is disabled, use the EXEC command **show interfaces**. An interface that has been shut down is shown as administratively down in the **show interfaces** command display.

Running CMCC Adapter Interface Loopback Diagnostics

The CMCC adapter does not provide software loopback support. You can use special loopback wrap plugs to perform hardware loopback with the ESCON and Parallel channel interfaces. Hardware loopback information is included in the hardware installation notes for the CMCC adapters.

Configuring a CMCC Adapter Core Dump

To obtain the output of a CMCC adapter core dump, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip domain-name` *name* `Router(config)# ip name-server` address `Router(config)# ip ftp username` name `Router(config)# ip ftp password` *password*	Configures the router FTP services.
Step 2	`Router(config)# exception slot` [*slot*] *protocol//:host/filename*	Configures the CMCC adapter core dump feature.

Note The exception slot command is only supported on the Cisco 7000 with RSP7000 and Cisco 7500 series routers. On the Cisco 7200 series routers, only FTP is supported.

While the router is running, you can use the **write** EXEC command to write the contents of a CMCC adapter that is not halted:

Command	Purpose
`Router# write`	Writes the contents of a CMCC adapter.

Note The output obtained by the **exception slot** command can be interpreted by a qualified Cisco technical support person.

CPA Microcode Load Configuration Examples

The following example shows output from running the **copy tftp** command to copy a new image to Flash memory:

```
Router#copy tftp:xcpa26-2 slot0:xcpa26-2

Address or name of remote host []? neptune
Translating "neptune"...domain server (10.20.30.10) [OK]
Destination filename [xcpa26-2]?
Accessing tftp://neptune/xcpa26-2...
Loading motto/xcpa26-2 from 10.20.30.10 (via Fast Ethernet0/0): !
  Expanding slot0:xcpa26-2_kernel_xcpa (343148 bytes):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_802 (237848 bytes):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_cmpc (319960 bytes):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_csna (89856 bytes): !!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_eca (461424 bytes):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_offload (80344 bytes): !!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_pca (69376 bytes): !!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_pseg_push (15936 bytes): !!!
  Expanding slot0:xcpa26-2_seg_tcpip (158896 bytes): !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  Expanding slot0:xcpa26-2_seg_tn3270 (601784 bytes):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 2387456/4774912 bytes]
2387456 bytes copied in 110.588 secs (21704 bytes/sec)
router#
```

After copying a CMCC ucode image to flash memory, a directory command of the flash device displays the following:

```
Router#dir slot0:

Directory of slot0:/
  1  -rw-          1   Aug 18 1998 12:29:12  xcpa26-2
  2  -rw-     344438   Aug 18 1998 12:29:12  xcpa26-2.kernel_xcpa
  3  -rw-     237848   Aug 18 1998 12:29:37  xcpa26-2.seg_802
  4  -rw-     319960   Aug 18 1998 12:29:56  xcpa26-2.seg_cmpc
  5  -rw-      89856   Aug 18 1998 12:30:15  xcpa26-2.seg_csna
  6  -rw-     461424   Aug 18 1998 12:30:20  xcpa26-2.seg_eca
  7  -rw-      80344   Aug 18 1998 12:31:03  xcpa26-2.seg_offload
  8  -rw-      69376   Aug 18 1998 12:31:07  xcpa26-2.seg_pca
  9  -rw-      15936   Aug 18 1998 12:31:11  xcpa26-2.seg_push
```

```
 10  -rw-      158896   Aug 18 1998 12:31:12  xcpa26-2.seg_tcpip
 11  -rw-      601784   Aug 18 1998 12:31:32  xcpa26-2.seg_tn3270
7995392 bytes total (5614116 bytes free)
```

The following example loads the microcode from an individual microcode image that is stored as a file in the PCMCIA card in slot 0:

```
Router(config)# microcode ecpa slot0:xcpa26-2
Router(config)# microcode reload
```

Configuring CLAW and TCP/IP Offload Support

This chapter provides information about configuring several related features on a Channel Interface Processor (CIP) or Channel Port Adapter (CPA) in a Cisco router to support TCP/IP environments. The features included in this chapter are Common Link Access to Workstation (CLAW), TCP/IP offload, and IP host backup.

This information is described in the following sections:

For a complete description of the commands in this chapter, refer to the "CLAW and TCP/IP Offload Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

For information on the CMPC+ feature, which also supports TCP/IP environments, refer to the "Configuring CMPC+" chapter in this publication.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page lv in the "Using Cisco IOS Software" chapter.

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

Overview of CLAW, TCP/IP Offload, and IP Host Backup Support

This section provides an overview of the TCP/IP environments supported by the CLAW, TCP/IP offload, and IP host backup. It includes the following topics:

TCP/IP Environments Using CLAW

TCP/IP mainframe protocol environments for IBM operating systems Multiple Virtual Storage (MVS) and Virtual Machine (VM) are supported. This support includes TCP/IP-based applications such as terminal emulation (Telnet), the File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP); and Network File System (NFS), a distributed file access system. In addition, Internet Control Message Protocol (ICMP) and User Datagram Protocol (UDP) are supported.

The Cisco IOS implements the CLAW channel protocol to transport data between the mainframe and a Cisco Mainframe Channel Connection (CMCC) adapter in TCP/IP environments. Each CLAW connection requires two devices out of a maximum of 256. Although this allows for a maximum of 128 CLAW connections per interface, a maximum of 32 CLAW connections per interface is recommended.

The CLAW packing feature enables the transport of multiple IP packets in a single channel operation and significantly increases throughput performance between a mainframe and a CMCC adapter. Currently, IBM's TCP/IP stack does not support the CLAW packing feature.

The CLAW packing feature requires changes to the mainframe CLAW driver support. In partnership with Cisco Systems, Interlink Computer Science (now Sterling Software) has made the corresponding CLAW driver change to Cisco IOS for S/390 Release 2 and Interlink TCPaccess 5.2. Customers must make the necessary changes to their host configurations in order to enable the CLAW packing feature.

TCP/IP Offload Environments

TCP/IP mainframe protocol environments for IBM operating systems MVS, VM, and Transaction Processing Facility (TPF) are supported.

The TCP/IP offload feature for CMCC adapters delivers the same function as the TCP/IP offload function on the IBM 3172 Interconnect Controller (Model 3), but with increased performance.

Offload Alias Support

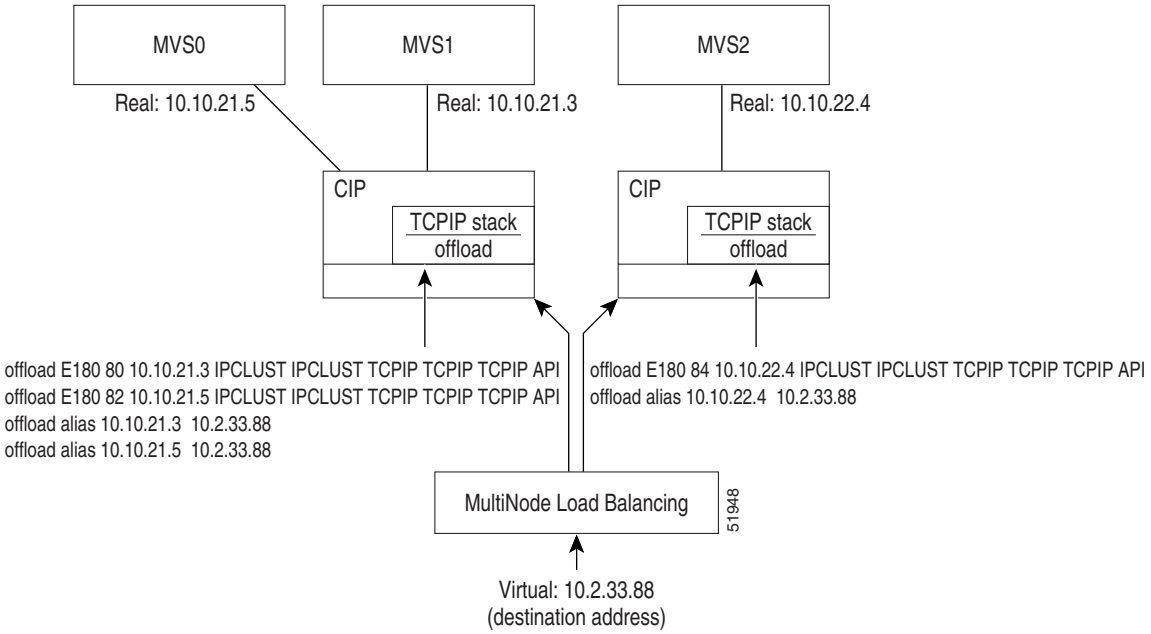
The Offload Alias feature allows multihomed IP addresses for offload devices. This feature enables dispatch-based load-balancing access to mainframe hosts through TCP/IP offload devices that are configured on a CMCC adapter. The Offload Alias feature supports load-balancing access to multiple hosts by allowing you to associate multiple real IP addresses with a virtual IP address at the offload device on a CMCC adapter. Each of the real IP addresses is associated with a common single virtual IP address, or alias, for client access.

Figure 249 shows an example of TCP/IP offload devices that are configured on two different CIP adapters, which are connected to multiple mainframe hosts using three different real IP addresses. The figure depicts the benefit that can be achieved when configuring offload alias support on a CMCC adapter in a networking environment that supports MultiNode Load Balancing (MNLB)—such as the MNLB Feature Set for LocalDirector.

The scenario shown in Figure 249 assumes that any of the configured hosts in the offload environment support the same services that a potential client seeks. A client specifies a virtual IP address, 10.2.33.88, as the destination address for those services. Using offload alias support, the virtual IP address 10.2.33.88 represents any of the three hosts (at real IP addresses 10.10.21.5, 10.10.21.3, or 10.10.22.4) as potential offload servers.

Upon receipt of the virtual IP address from a client, a networking architecture that supports MNLB can assess the load at any of the associated real IP addresses of the hosts supported by the offload devices. Based upon the load information, the load-balancing software forwards the packet to a particular real IP address. The destination IP address within the packet always appears as the virtual IP, or alias, address.

Figure 249 Two Offload Devices Using the Same Virtual IP Address to Access Multiple Real IP Addresses



IP Host Backup Environments

You can connect multiple mainframes to a single CMCC adapter using ESCON channel attachment. Often these mainframes run using the ESCON Multiple Image Facility (EMIF), which permits the physical machine to be divided into multiple logical partitions (LPARs). By defining an unused partition on another mainframe, a user can move the operating system from a failed mainframe or mainframe partition to the unused partition. By having multiple paths to each device, the move is accomplished without changing the mainframe software. This function also permits moving an IP stack between multiple operating system images.

On the CMCC adapter, each IP connection is treated as a physical device. The CMCC adapter does not support multiple active paths to a single IP connection (or device). Prior to IP Host Backup, the router configuration had to be changed whenever the mainframe operating system was moved from one mainframe or LPAR to another. The IP Host Backup feature permits the mainframe operating system to be moved from one mainframe to another without requiring a change to the router configuration at the time of the move.

Note The IP Host Backup feature does not provide single system image or automatic failover to a waiting backup application. Host operator action on the mainframe is required in these instances.

Preparing to Configure CLAW, TCP/IP Offload and IP Host Backup Support

The following topics in this section provide information that is useful when you are planning to configure CLAW, TCP/IP offload, or IP host backup support:

- Hardware and Software Requirements, page 4
- Mainframe Host Configuration Considerations, page 5

Hardware and Software Requirements

This section provides information about the router and mainframe requirements to support CLAW, TCP/IP offload, or IP host backup support. The router requirements are the same to support all features.

Router Requirements

The CLAW, TCP/IP offload, and IP host backup features are supported on the following router platforms:

- Cisco 7500 series—Supports CIP adapters
- Cisco 7200 series—Supports the ECPA and PCPA adapters
- Cisco 7000 series with RSP7000—Supports CIP adapters

You must configure CLAW, TCP/IP offload, and IP host backup features on the physical interface of a CMCC adapter. For a CIP, the physical interface is either 0 or 1. For the CPA adapters, ECPA and PCPA, the physical interface is port 0.

Mainframe Requirements

The CMCC adapters provide support for IP clients to communicate with TCP/IP stacks on IBM hosts that are running the following operating systems:

- IBM Multiple Virtual Storage (MVS)
- OS/390
- Virtual Machine (VM)
- Transaction Processing Facility (TPF)—Supported by the TCP/IP offload feature only

Note OS/390 Version 2.5 and later does not support the TCP/IP offload feature on a CMCC adapter.

TCP/IP protocol services are supported by the following TCP/IP software on the host:

- IBM TCP/IP for MVS Version 2, Release 2.1 and later
- IBM TCP/IP for VM Version 2, Release 2 and later
- Cisco IOS for S/390 Release 1 and later—Interlink Computer Sciences TCP access for MVS

Note Certain Authorized Program Analysis Reports (APARs) and Program Temporary Fixes (PTFs) are required with the IBM TCP/IP software on MVS and VM including: PTF UN47759 (and later) and PTF UN51632 (APAR PN45287).

Mainframe Host Configuration Considerations

Configuring CLAW, TCP/IP offload, or IP host backup support requires that you perform tasks for configuration of the mainframe and the router sides of the network environment.

Often in the mixed network environment of mainframes and LANs, a systems programmer installs and maintains the mainframe side of the network, while a network engineer manages the routers on the LAN side of the network. In such an environment, the successful configuration of CLAW, TCP/IP offload, or IP host backup support requires the close coordination between these job functions at a customer site.

This chapter contains information for both the network engineer and the host systems programmer to properly configure the network devices for CLAW, TCP/IP offload, and IP host backup support. The tasks for configuring this support are organized by whether they are host-related configuration tasks or router-related configuration tasks. In addition, a topic for correlating the mainframe and router configuration is provided so that you can identify the dependencies between the host and router configuration elements and be sure that they are set up correctly.

Defining the Channel Subsystem for the Router

To establish the path and allocate the range of subchannel addresses that the CMCC adapter can use for the CLAW, TCP/IP offload, or IP host backup features, you need to specify the channel subsystem definitions in the Input/Output Control Program (IOCP) or Hardware Configuration Definition (HCD).

For more information about the statements that might be defined in an IOCP file for parallel channels and ESCON channels on the CIP or CPA, see the “Defining the Channel Subsystem for the Router” section in the “Configuring Cisco Mainframe Channel Connection Adapters” chapter of this guide.

Disabling the Missing Interrupt Handler

Because the appropriate configuration of the missing interrupt handler (MIH) varies according to the protocols and software releases used, Cisco offers the following guidance:

- For OS/390 releases Version 2 Release 4 and earlier, set the MIH to zero.
- For OS/390 releases later than Version 2 Release 4 and z/OS releases, refer to the following section of the z/OS Communications Server IP Configuration Reference: http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/f1a1b420/1.2.13?SHELF=f1a1bk31&DT=20020604120755#HDRMOLLY

For information about how to disable the MIH for the unit addresses being used for your CMCC adapter configuration, see the section "Disabling the Missing Interrupt Handler" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter of this publication.

Related Publications

The following mainframe-related publications might be useful when configuring TCP/IP on the mainframe to support CLAW, TCP/IP offload, and IP host backup support on a CMCC adapter in the router:

- *OS/390 eNetwork Communications Server: IP Planning and Migration Guide*, SC31-8512-02
- *OS/390 eNetwork Communications Server: IP Configuration Guide*, SC31-8513-02
- *OS/390 eNetwork Communications Server: IP User's Guide*, GC31-8514-02
- *OS/390 eNetwork Communications Server: IP Diagnosis*, SC31-8521-02
- *OS/390 eNetwork Communications Server: IP and SNA Codes*, SC31-8571-02

PROFILE.TCPIP Host Configuration Task List

Note The information in this section on configuring the PROFILE.TCPIP data set on the host is provided to help you correlate the host and router configurations. For the latest information about configuring TCP/IP support on the mainframe, refer to your host publications.

Complete the following tasks on the host to properly configure the PROFILE.TCPIP data set to support the CLAW feature:

- Defining the Device Statement, page 7
- Defining the Link Statement for CLAW, page 8
- Defining the Link Statements for Offload, page 8
- Defining the Home Statement for CLAW, page 9
- Defining the Gateway Statement, page 9
- Defining the Defaultnet Statement, page 10

- Defining the Start Statement, page 10
- Configuring Dynamic Routing, page 10

Consider the following sample configuration statements when you perform the tasks to configure the PROFILE.TCPIP data set on the host. The names shown in bold and italic identify the relationship to some of the important configuration names within the PROFILE.TCPIP configuration.

Sample PROFILE.TCPIP Configuration for CLAW

```
; Device statement—Defines each CIP subchannel. Values CLAW and NONE
; are required. The read-size argument must be 4096.
;
; DEVICE device-name CLAW subchannel-address host-name device-name-on-router NONE
; read-buffers write-buffers read-size write-size
;
DEVICE CIP1 CLAW 762 CISCOVM CIP NONE 20 20 4096 4096
;
; Link statement—Defines a link name and type. Values "IP" and "0"
; are required for CLAW
;
; LINK link-name IP 0 device-name
;
LINK CIPL1 IP 0 CIP1
;
; Home statement—Specifies an IP address for a link
;
; HOME ip-address link-name

HOME 172.16.20.2 CIPL1
;
!
; Routing information (if you are not using the ROUTED SERVER)
GATEWAY
; NETWORK FIRST HOP  DRIVER    PCKT_SZ    SUBN_MSK       SUBN_VALUE
;
172.16      =         CIPL1     4096       0.0.255.0       0.0.20.0
DEFAULTNET 172.16.20.1 CIPL1    1500       0
!
;START statements—Starts the IP datagram (CLAW) device
;
; START device-name
;
START CIP1
!
```

Figure 249 shows the corresponding claw configuration on the router.

Sample CLAW Configuration on the Router

```
claw C010 62 172.16.20.2 CISCOVM CIP TCPIP TCPIP
```

See the "CLAW and TCP/IP Offload Support Configuration Examples" section on page 24 for more examples.

Defining the Device Statement

The DEVICE statement in the PROFILE.TCPIP data set defines the link type (CLAW) and a user-specified name for the CLAW device according to the following format:

```
DEVICE device-name CLAW subchannel-address host-name device-name-on-router NONE
read-buffers write-buffers read-size write-size
```

When you configure the DEVICE statement consider the following points:

- The user-specified *device-name* argument does not have any dependencies with the router configuration.
- The name following the DEVICE keyword must match the *device-name* argument in the corresponding LINK statement in the PROFILE.TCPIP data set.
- The name that you specify for the *device-name-on-router* argument must match the device name that you specify in the **claw** configuration command in the router. This is "CIP" in Figure 249 and Figure 249.
- The CIP sends blocks up to 4096 in size. If the value for the read size is any smaller, you might have overruns, so the *read-size* argument must be 4096. The *write-size* argument should be less than or equal to 4096.

Defining the Link Statement for CLAW

The LINK statement for CLAW in the PROFILE.TCPIP data set specifies the link type and a name for the link according to the following format:

```
LINK link-name IP 0 device-name
```

When you configure the LINK statement for CLAW consider the following points:

- The user-specified *link-name* argument does not have any dependencies with the router configuration.
- "IP" and "0" are required values for CLAW.
- The name that you specify for the *device-name* argument in the LINK statement must match the *device-name* argument in the DEVICE statement. In Figure 249 this name is CIP1.

Defining the Link Statements for Offload

The LINK statements for offload support in the PROFILE.TCPIP data set specify the link type and a name for the link and the IP address of the TCP/IP stack in the router according to the following format:

```
LINK link1-name OFFLOADLINK1 1 device-name

LINK link2-name OFFLOADAPIBROAD ip-address device-name link1-name
```

Note The HOME statement is replaced by the second LINK statement in TCP/IP offload configuration.

When you configure the first LINK statement for offload consider the following points:

- The user-specified *link1-name* argument does not have any dependencies with the router configuration.
- "OFFLOADLINK1" and "1" are required values for offload support.
- The name that you specify for the *device-name* argument in the LINK statement must match the *device-name* argument in the DEVICE statement.

When you configure the second LINK statement for offload consider the following things:

- The user-specified *link2-name* argument does not have any dependencies with the router configuration.
- "OFFLOADAPIBROAD" is a required keyword.
- The *ip-address* argument in the HOME statement must match the *ip-address* argument that you specify in the **claw** configuration command in the router.
- The name that you specify for the *device-name* argument in the LINK statement must match the *device-name* argument in the DEVICE statement.
- The *link1-name* argument must match the link name in the first LINK statement.

Defining the Home Statement for CLAW

The HOME statement in the PROFILE.TCPIP data set specifies an IP address for a link according to the following format:

```
HOME ip-address link-name
```

When you configure the HOME statement consider the following points:

- The HOME statement is not used when configuring TCP/IP offload support.
- The *ip-address* argument in the LINK statement must match the *ip-address* argument that you specify in the **claw** configuration command in the router. This is "172.16.20.2" in Figure 249 and Figure 249.
- The *link-name* argument in the HOME statement must match the *link-name* argument in the LINK statement. In Figure 249 this name is CIPL1.

Defining the Gateway Statement

The GATEWAY statement in the PROFILE.TCPIP data set defines a static route from the host to the router according to the following format:

```
GATEWAY network first-hop driver packet-size subnet-mask subnet-value
```

When you configure the GATEWAY statement consider the following points:

- Specify the value "=" in the *first-hop* argument to indicate the local network.
- The *driver* argument in the GATEWAY statement for CLAW must match the *link-name* in the LINK statement. In Figure 249 this name is CIPL1.
- The *driver* argument in the GATEWAY statement for offload must match the *link2-name* in the second LINK statement.
- The GATEWAY statement is not used if the host is using dynamic routing. For information about configuring dynamic routing, see the "Configuring Dynamic Routing" section on page 10.

Defining the Defaultnet Statement

The DEFAULTNET statement in the PROFILE.TCPIP data set defines the default gateway for the network according to the following format:

```
DEFAULTNET first-hop driver packet-size subnet-mask subnet-value
```

When you configure the DEFAULTNET statement consider the following points:

- Specify the IP address for the CMCC adapter in the *first-hop* argument.
- The *driver* argument in the GATEWAY statement must match the *link-name* in the LINK statement. In Figure 249 this name is CIPL1.

Defining the Start Statement

The START statement instructs the host to start the CLAW device when TCP/IP is started according to the following format:

```
START device-name
```

When you configure the START statement consider the following points:

- The *device-name* argument in the START statement must match the *device-name* in the DEVICE statement. In Figure 249 this name is CIP1.
- You must have a START statement for each device that you want TCP/IP to start.

Configuring Dynamic Routing

The BSDROUTINGPARMS statement in the PROFILE.TCPIP data set allows you to configure the host to support dynamic routing algorithms using Routing Information Protocol (RIP) with ROUTED or OROUTED.

Note Do not configure BSDROUTINGPARMS for OMPROUTE, which uses the Open Shortest Path First (OSPF) Protocol.

Configure the BSDROUTINGPARMS statement according to the following format:

```
BSDROUTINGPARMS [TRUE|FALSE]
link  maxmtu  metric  subnet-mask  dest-addr
ENDBSDROUTINGPARMS
```

The following example shows a dynamic routing configuration with the router IP address specified as a point-to-point link:

```
BSDROUTINGPARMS TRUE
CIPL1 4096 0 255.255.255.0 172.16.21.1
ENDBSDROUTINGPARMS
```

When you configure BSDROUTINGPARMS consider the following points:

- Do not configure the GATEWAY statement in PROFILE.TCPIP when you configure BSDROUTINGPARMS.
- The BSDROUTINGPARMS statement supports the following options:
  - TRUE—Maximum transmission unit (MTU) for the interface is specified by the *maxmtu* argument.
  - FALSE—Offnet traffic has an MTU size of 576 bytes.
- The *link-name* matches the *link-name* argument in the LINK statement.
- The *maxmtu* specifies the maximum MTU in bytes for the network.
- The *metric* is 0 to indicate that the network is directly connected to the host.
- The *subnet-mask* is the bit mask associated with the subnet for the link.
- The *dest-addr* specifies point-to-point links. Specify a value of "0" if the link is on a broadcast-capable network.

Configuring CLAW Support

The CLAW feature in Cisco IOS software implements the CLAW channel protocol to transport data between the mainframe host and a CMCC adapter in TCP/IP environments. This section describes the configuration tasks required to enable CLAW support on the mainframe and router and includes the following topics:

- CLAW Configuration Guidelines, page 11
- CLAW Router Configuration Task List, page 12

CLAW Configuration Guidelines

To configure the CLAW feature, you must configure the TCP/IP stack on the host and the CMCC adapter. Consider the following guidelines as you prepare to configure CLAW support:

- To define the host subchannel (or path) and device, use the **claw** command on the router. The **claw** command is configured on the CMCC adapter's physical interface (port 0 or 1 on a CIP; port 0 on a CPA).
- The host IOCP or HCD parameters must define a subchannel pair for use by the CLAW device on a CMCC adapter. The even address is used for host reads and the odd address is used for host writes.
- The PROFILE.TCPIP data set DEVICE statement on the host must specify the CLAW subchannel address defined in the host IOCP or HCD configuration.
- The host IOCP or HCD parameters must coordinate with the **claw** command parameters on the router to specify the subchannel path, device, and subchannel address.
- The PROFILE.TCPIP data set parameters on the host must coordinate with the **claw** command parameters on the router.
- Up to a maximum of 32 CLAW connections per interface are recommended.
- CLAW can coexist with TCP/IP offload, IP host backup, CSNA, CMPC, CMPC+ and TN3270 server features on the router.

CLAW Router Configuration Task List

The following sections describe how to configure a CMCC adapter for CLAW support. These tasks require configuration of the physical port on a CMCC adapter:

- Configuring the CLAW Subchannels, page 12
- Assigning an IP Address to the Network Interface, page 14
- Configuring Other Interface Support, page 14

Refer to the "CLAW and TCP/IP Offload Support Configuration Examples" section on page 24 to see an example of CLAW configurations.

Configuring the CLAW Subchannels

Configuring the CLAW subchannels establishes the physical path between the CMCC interface and the mainframe channel. You must define the devices, or tasks, supported on the interface. Some of the information that you need to perform this task is derived from the following host system configuration files: Channel subsystem definitions such as the IOCP or HCD, and the TCP/IP configuration.

The CLAW Packing feature requires changes to the mainframe CLAW driver support. In partnership with Cisco Systems, Interlink Computer Science has made the corresponding CLAW driver change to Cisco IOS software for S/390 Release 2 and Interlink TCPaccess 5.2. Configuration parameters in the host TCP/IP applications must change to enable the CLAW Packing feature.

To define a host subchannel supported by the CLAW feature, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure CLAW support and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*Port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*Port* value corresponds to port 0.

	Command	Purpose
Step 2	`Router(config-if)# claw path device-address ip-address host-name device-name host-app device-app [broadcast] [backup]`	Defines the CLAW subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. • *ip-address*—IP address of the host. This address is specified in the HOME statement of the host TCP/IP configuration file. • *host-name*—Name of the host. This name is specified in the DEVICE statement of the host TCP/IP configuration file. • *device-name*—Name for the router's CMCC device. This name is specified in the DEVICE statement of the host TCP/IP configuration file. • *host-app*—Host application name as specified in the host application file. If connected to the host using TCP/IP without CLAW packing, this value is **tcpip**. If attached to other applications, this value must match the value specified in that host application. To enable CLAW packing, this value is **packed**. • *device-app*—CLAW workstation application name as specified in the host application file. If connected to the host using TCP/IP without CLAW packing, this value is **tcpip**. If attached to other applications, this value must match the value specified in that host application. To enable CLAW packing, this value is **packed**. The available options for this command are: • **broadcast**—(Optional) Enables broadcast forwarding to the host on the subchannel (for RIP updates). • **backup**—(Optional) Enables the CLAW connection to be used as part of a backup group of CLAW connections for the specified IP address.

Use the **no claw** command to remove the CLAW subchannel device.

Mainframe Configuration Tips

- Configuring the subchannel information in the router requires that you correlate the *path* and *device* information from the IOCP or HCD file on the host.
  - The *path* argument is a four-digit hexadecimal value that concatenates the path value (2 digits), EMIF partition number (1 digit), and control unit logical address (1 digit).
  - The *device* argument is a valid number in the UNITADD range of the IOCP CNTLUNIT statement.

  For detailed information about how to determine the *path* and *device* values for the **claw** command, see the "Correlating Channel Configuration Parameters" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.
- Configuring the subchannel information in the router also requires that you correlate the *ip-address*, *host-name*, *device-name*, *host-app*, and *device-app* arguments with the TCP/IP configuration on the host. For a summary of how the mainframe and router options correlate, see the "Correlating the Router and Mainframe Configuration Elements" section on page 22.

Assigning an IP Address to the Network Interface

You must assign an IP address to the channel interface so that it can communicate with other devices (or tasks) on the network. The IP address you assign to the interface must be in the same subnetwork as the hosts with which you wish to communicate.

To assign an IP address to the network interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# ip address ip-address mask [secondary]`	Assigns an IP address to the network interface.

Configuring Other Interface Support

To enhance the performance of the CLAW support, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the CLAW interface on which to configure the options and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*Port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*Port* value corresponds to port 0.

Step 2	`Router(config-if)# ip mtu 4096`
	
	or
	
	`Router(config-if)# ip mtu 4092`
		Sets the MTU size of IP packets sent on the interface to 4096 bytes. The default MTU is 4472 bytes across the channel on a CMCC. The CLAW feature cannot accept packets larger than 4096 bytes on a CMCC.
		
		or
		
		Sets the MTU size of IP packets sent on the interface to 4092 bytes when running CLAW packing. The default MTU is 4472 bytes across the channel on a CMCC. The CLAW feature cannot accept packets larger than 4092. It cannot be 4096 due to the presence of a four byte length field that preceeds the packet when running CLAW packing.

Step 3	`Router(config-if)# ip route-cache same-interface`	(Optional) Includes fast switching support for multiple IP datagram applications running on the same CMCC adapter.
Step 4	`Router(config-if)# no ip redirects`	(Required when configuring host-to-host communications through the same interface) Disables the sending of ICMP redirect messages if the Cisco IOS software is forced to resend a packet through the same interface on which was received.

For additional information about configuring other CMCC adapter options such as the data rate for parallel channel interfaces, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.

Configuring TCP/IP Offload Support

The TCP/IP Offload feature in Cisco IOS software implements TCP/IP processing on the router using the CLAW protocol. This section describes the configuration tasks required to enable TCP/IP offload support on the router and includes the following topics:

- TCP/IP Offload Configuration Guidelines, page 15
- TCP/IP Offload Router Configuration Task List, page 17

TCP/IP Offload Configuration Guidelines

To configure the TCP/IP Offload feature, you must configure the TCP/IP stack on the host and the CMCC adapter. Consider the following guidelines as you prepare to configure TCP/IP offload support:

- To define the host subchannel (or path) and device, use the **offload** command on the router. The **offload** command is configured on the CMCC adapter's physical interface (port 0 or 1 on a CIP; port 0 on a CPA).
- The host IOCP or HCD parameters must define a subchannel pair for use by the offload device on a CMCC adapter. The even address is used for host reads and the odd address is used for host writes.
- The PROFILE.TCPIP data set DEVICE statement on the host must specify the offload subchannel address defined in the host IOCP or HCD configuration.
- The host IOCP or HCD parameters must coordinate with the **offload** command parameters on the router to specify the subchannel path, device, and subchannel address.
- The PROFILE.TCPIP data set parameters on the host must coordinate with the **offload** command parameters on the router.
- TCP/IP offload uses CLAW and can coexist with IP host backup, CMPC, CMPC+ and TN3270 server features on the router. It is important to be sure that the SAPs used for these entities are unique.

TCP/IP Offload Router Configuration Task List

The following sections describe how to configure a CMCC adapter for TCP/IP offload support. These tasks require configuration of the physical port on a CMCC adapter:

- Configuring the Offload Subchannels, page 17
- Assigning an IP Address to the Network Interface, page 19
- Configuring Offload Alias Support, page 19
- Configuring Other Interface Support, page 20

Refer to the "CLAW and TCP/IP Offload Support Configuration Examples" section on page 24 to see an example of an offload configuration.

Configuring the Offload Subchannels

Configuring the offload subchannels establishes the physical path between the CMCC interface and the mainframe channel. You must define the devices, or tasks, supported on the interface. Some of the information that you need to perform this task is derived from the following host system configuration files: Channel subsystem definitions such as the IOCP or HCD, and the TCP/IP configuration.

To define a host subchannel supported by the offload feature, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure offload support and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# offload path device-address ip-address host-name device-name host-ip-link device-ip-link host-api-link device-api-link [broadcast][backup]`	Defines the offload subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. • *ip-address*—IP address of the host. This address is specified in the second LINK statement of the host TCP/IP configuration file. • *host-name*—Name of the host. This name is specified in the DEVICE statement of the host TCP/IP configuration file. • *device-name*—Name for the router's CMCC device. This name is specified in the DEVICE statement of the host TCP/IP configuration file. • *host-ip-link*—CLAW host link name as specified by the host application. For IBM VM and MVS TCP/IP stacks, this value is always **tcpip**. • *device-ip-link*—CLAW workstation link name as specified by the host application. For IBM VM and MVS TCP/IP stacks, this value is always **tcpip**. • *host-api-link*—CLAW host link name for the application programming interface (API) link as specified by the host application. For IBM VM and MVS TCP/IP stacks, this value is always **tcpip**. • *device-api-link*—Offload link name for the API link as specified by the host application. For IBM VM and MVS TCP/IP stacks, this value is always **api**. The available options for this command are: • **broadcast**—(Optional) Enables broadcast forwarding to the host on the subchannel (for RIP updates). • **backup**—(Optional) Enables the offload connection to be used as part of a backup group of offload connections for the specified IP address.

Use the **no offload** command to remove the offload subchannel device.

Mainframe Configuration Tips

- Configuring the subchannel information in the router requires that you correlate the *path* and *device* information from the IOCP or HCD file on the host.
  - The *path* argument is a four-digit hexadecimal value that concatenates the path value (two digits), EMIF partition number (one digit), and control unit logical address (one digit).
  - The *device* argument is a valid number in the UNITADD range of the IOCP CNTLUNIT statement.

  For detailed information about how to determine the *path* and *device* values for the **offload** command, see the "Correlating Channel Configuration Parameters" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.
- Configuring the subchannel information in the router also requires that you correlate the *ip-address*, *host-name*, and *device-name* arguments with the TCP/IP configuration on the host. For a summary of how the mainframe and router options correlate, see the "Correlating the Router and Mainframe Configuration Elements" section on page 22.

Assigning an IP Address to the Network Interface

You must assign an IP address to the channel interface so that it can communicate with other devices (or tasks) on the network. The IP address you assign to the interface must be in the same subnetwork as the hosts with which you wish to communicate.

To assign an IP address to the network interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# ip address` *ip-address mask* `[secondary]`	Assigns an IP address to the network interface.

Configuring Offload Alias Support

The Offload Alias feature allows you to assign up to eight virtual IP addresses to a single real IP address for an offload device. You must configure TCP/IP offload support before configuring support for offload aliases. Offload aliases are configured on the physical interface of a CMCC adapter.

To configure offload alias support on a CMCC adapter, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# offload alias` *real-ip alias-ip*	Assigns a virtual IP address (alias) to the real IP address of an offload device.

Use the **no offload alias** command to remove the alias IP address.

Verifying Offload Alias Configuration

To configure and verify offload alias support on a CMCC adapter, perform the following steps:

Step 1 Configure offload and offload alias support on the CMCC adapter as shown in the following example:

```
interface channel 3/1
 ip address 10.10.21.1 255.255.255.0
 offload E180 83 10.10.21.3 IPCLUST IPCLUST TCPIP TCPIP TCPIP API
 offload alias 10.10.21.3 10.2.33.88
```

Step 2 Create a static route from the alias IP address to the real IP address as shown in the following example:

```
ip route 10.2.33.88 255.255.255.255 10.10.21.3
```

Step 3 Run a server, such as Telnet, on the host supported by the offload device.

Step 4 From a client device, run Telnet to the host using the alias IP address, which is 10.2.33.88 in this configuration example.

Configuring Other Interface Support

To enhance the performance of offload support, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel` *slot/port*	Selects the CLAW interface on which to configure the options and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# ip mtu 4096`	(Recommended for CPA interfaces) Sets the MTU size of IP packets sent on the interface to 4096 bytes. The default MTU is 4472 bytes across the channel on a CPA. Specifying 4096 eliminates fragmentation and increases performance.
Step 3	`Router(config-if)# ip route-cache same-interface`	(Optional) Includes fast switching support for multiple IP datagram applications running on the same CMCC adapter.
Step 4	`Router(config-if)# no ip redirects`	(Required when configuring host-to-host communications through the same interface) Disables the Internet Control Message Protocol (ICMP) flow that notifies the router of a better path when frames are being sent from one host to another host through the router.

For additional information about configuring other CMCC adapter options such as the data rate for parallel channel interfaces, see the “Configuring Cisco Mainframe Channel Connection Adapters” chapter in this publication.

IP Host Backup Support Configuration Task List

The following sections describe how to configure a CMCC adapter for IP Host Backup support. With IP Host Backup, you can configure a backup group for each CLAW or offload device, one path at a time, or you can specify a group of IP host paths and then configure which CLAW or offload IP addresses are used with those paths. Using the second method, specifying paths, provides a shortcut to the one at a time method.

Configuring a CLAW IP Host Backup Group

To configure the CLAW IP Host Backup, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# claw path device-address ip-address host-name device-name host-app device-app [broadcast] backup`	Defines the CLAW subchannel device for backup support.

For more information about configuring the **claw** command, see the "Configuring CLAW Support" section on page 11.

Configuring an Offload IP Host Backup Group

To configure the Offload IP Host backup, use the following command in interface configuration mode to configure an Offload device (read and write subchannel) for communication with a mainframe TCP/IP stack in Offload mode:

Command	Purpose
`Router(config-if)# offload path device-address ip-address host-name device-name host-ip-link device-ip-link host-api-link device-api-link [broadcast] backup`	Defines the offload subchannel device for backup support.

For more information about configuring the **offload** command, see the "Configuring TCP/IP Offload Support" section on page 15.

Configuring an IP Host Backup Group Using Paths

You can define a backup group by specifying a path, or group of paths, that are used as the IP Host Backup. Under the backup group, you can have multiple backup connections defined that all use the same IP address.

To configure the IP Host Backup using paths, use the following commands beginning in interface configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the IP host backup paths and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# path path [[path ...]]`	Defines the backup path or paths for this group and enters IP Host Backup configuration mode.
Step 3	`Router(config-if-path)# claw device-address ip-address host-name device-name host-app device-app [broadcast]`	Defines the CLAW parameters for this device.
Step 4	`Router(config-if-path)# offload device-address ip-address host-name host-ip-link device-ip-link host-api-link device-api-link [broadcast]`	Defines the offload parameters for this device.
Step 5	`Router(config-if-path)# exit`	Exits IP Host Backup configuration mode and returns to interface configuration mode.

Correlating the Router and Mainframe Configuration Elements

Table 14 shows a summary of the configuration elements on the router and host that must be correlated for proper operation of CLAW and TCP/IP offload support. The column labeled “Configuration Element” identifies the type of entity to be configured. The columns labeled “Router Configuration” and “Mainframe Configuration” identify the related parameters on the router and the mainframe whose values must be compatible or match.

Table 14 Relationship of Router and Mainframe Configuration Elements

Configuration Element	Router Configuration	Mainframe Configuration
Subchannels	*path* and *device* arguments of the **claw** and **offload** commands	RESOURCE PARTITION, CHPID, and CNTLUNIT statements of the IOCP definition defining the following parameters for the CLAW or offload channel path: • LPAR number (if defined) in the RESOURCE PARTITION and CHPID statements—Specify in the third digit of the *path* argument in the router **claw** and **offload** commands. • CUADD value (if defined) in the CNTLUNIT statement—Specify in the fourth digit of the *path* argument in the router **claw** and **offload** commands. • Available device address in the UNITADD parameter of the CNTLUNIT statement—Specify in the *device* argument of the router **claw** and **offload** commands.
Host IP address	*ip-address* argument of the **claw** and **offload** commands	• For CLAW—*ip-address* argument in the HOME statement of the PROFILE.TCPIP data set. • For offload—*ip-address* in the second LINK statement of the PROFILE.TCPIP data set.
Host name	*host-name* argument of the **claw** and **offload** commands	*host-name* argument in the DEVICE statement of the PROFILE.TCPIP data set.
Device name	*device-name* argument of the **claw** and **offload** commands	*device-name* argument in the DEVICE and statement of the PROFILE.TCPIP data set.

Monitoring and Maintaining CLAW and TCP/IP Offload

To monitor CMCC adapter interface status, you can display information about the interface, including the version of the software and the hardware, the controller status, and statistics about the interfaces. In addition, you can display information about feature-related statistics on the CMCC adapter. This section lists some additional commands that are useful when monitoring CMCC adapter interfaces that are configured for TCP/IP environments.

For a complete list of the **show** commands that are related to monitoring CMCC adapter interfaces, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication. To display the full list of **show** commands, enter **show ?** at the EXEC prompt.

To display information related to CLAW and TCP/IP offload configurations, use the following commands in EXEC mode:

Command	Purpose
Router# **show extended channel** *slot/port* **backup** [*ip-address*]	Displays information about CLAW and offload commands for each backup group.
Router# **show extended channel** *slot/port* **icmp-stack** [*path* [*ip-address*]]	Displays information about the ICMP stack running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **ip-stack** [*ip-address*]	Displays information about the IP stack running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **packing names** [*path* [*device*]]	Displays CLAW packing names and their connection state.
Router# **show extended channel** *slot/port* **packing stats** [*path*[*device*]]	Displays CLAW packing statistics.
Router# **show extended channel** *slot/port* **tcp-connections** [[**loc-ip-addr**[*loc-port*[*rem-ip-addr*[*rem-port*]] [**detail**|**summary**]	Displays information about the TCP sockets on a channel interface that is configured for offload.
Router# **show extended channel** *slot/port* **tcp-stack** [*ip-address*]	Displays information about the TCP stack running on the CMCC adapter interfaces that are configured for offload.

CLAW and TCP/IP Offload Support Configuration Examples

The following sections include examples to help you understand different aspects of interface configuration:

IP Address and Network Mask Configuration Example

The following example assigns an IP address and network mask to the IBM channel attach interface on the router:

```
ip address 197.91.2.5 255.255.255.0
```

CLAW Configuration Example

The following example configures the IBM channel attach interface to support a directly connected device:

```
claw 0100 00 197.91.2.2 VMSYSTEM C7000 TCPIP TCPIP
```

CLAW Packing Configuration Example

The following example configures the IBM channel attach interface to support CLAW packing on HOSTA and HOSTC and the nonpacked version of CLAW on HOSTB:

```
interface Channel0/0
ip mtu 4092
  ip address 172.18.4.49 255.255.255.248
  no keepalive
  claw C010 F2 172.18.4.50 HOSTA RTRA PACKED PACKED
  claw C020 F4 172.18.4.52 HOSTB RTRA TCPIP TCPIP
  claw C030 F6 172.18.4.53 HOSTC RTRA PACKED PACKED
```

The following example shows a CLAW definition in the host configuration file for Cisco IOS/390:

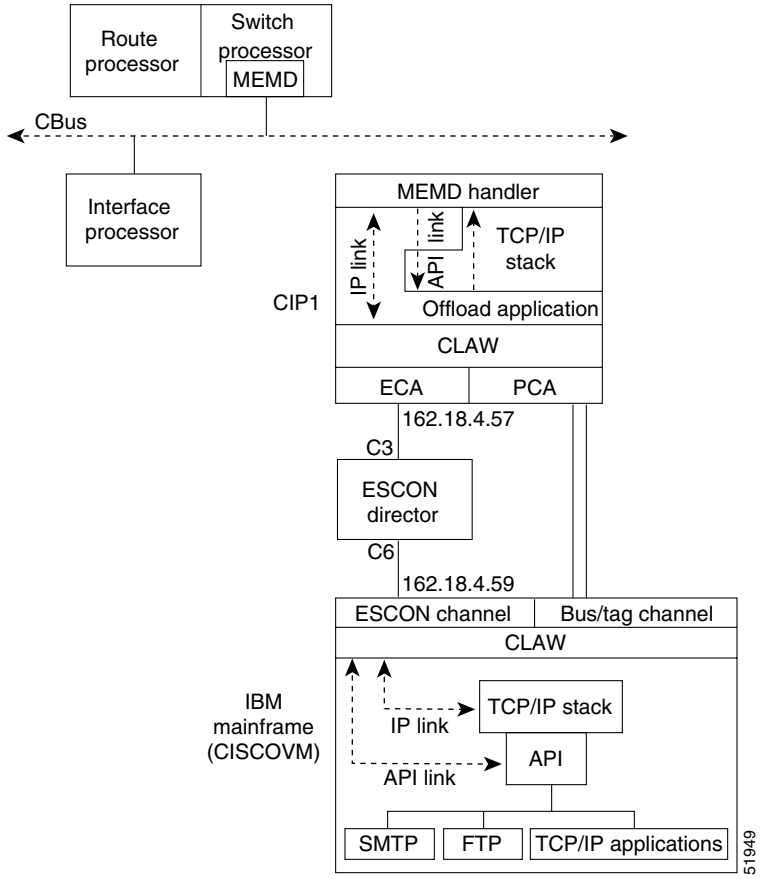
```
000100 *-------------------------------------------
000200 * Member: IOS390R2.V510.PARM(TCPCFGxx)
000300 * Description: TCP task group configuration
000400 *-------------------------------------------
000500
000600 * Define the virtual medium
000700
000800 MEDIA VIRTUAL MTU(4096) NAME(LOOPBACK)
000900
001000 * Define the physical medium
001100
001200 MEDIA CLAW    MTU(4096) NAME(ROGCLAW) ASSIST
001300
001400 * Define the host
001500
001600 NETWORK IPADDRESS(172.18.4.50)
001700      SUBNET(255.255.255.248)
001800
001900 *
002000
002100 CLAW DEVADDR(8f2)
002200      BUFSIZE(32768)
002300      IBUF(5)
002400      OBUF(5)
002500      RESTART(60)
002600      HOSTNAME(HOSTA)
002700      WSNAME(RTRA)
002800      START
002900      PACKED
003000
003100 * Define gateway
003200
003300 ROUTE DEST(0.0.0.0) ROUTE(172.18.4.49)
003400
003500 * Define the transport pr
003600
003700 TCP  MAXRCVBUF(131072)
003800      MAXSNDBUF(131072)
003900      DEFRCVBUF(131072)
004000      DEFSNDBUF(131072)
004100      DELAYACK(2)
004200      FASTRX(3)
004300      MAXRXMIT(18)
004400      MINDEV(90)
004500      PORTUSE(1:4095)
004600      PORTASGN(4096:8191)
004700
```

```
004800 UDP   MAXRCVBUF(64000)005200       PORTUS
004900       MAXSNDBUF(64000)
005000       DEFRCVBUF(64000)
005100       DEFSNDBUF(64000)005300       PORTAS
005200       PORTUSE(1:4095)
005300       PORTASGN(4096:8191)
005400
005500 RAW    MAXRCV
005600       MAXSND
005700
```

TCP/IP Offload Configuration Example

The following example consists of the mainframe host profile statements, buffer poolsize recommendations, and router configuration statements for the network shown in Figure 250.

Figure 250 Offload Network Configuration Block Diagram



Host Profile Statements

```
; Device statement
DEVICE OFF CLAW 762 CISCOVM CIP1 NONE 20 20 4096 4096
!
; Link Statements (both needed)
LINK OFFL OFFLOADLINK1 1 OFF
LINK MEMD OFFLOADAPIBROAD 162.18.4.59 OFF OFFL
```

```
!
; Home Statement
; (No additional home statements are added for offload)
!
!
; Routing information (if you are not using the ROUTED SERVER)
GATEWAY
; NETWORK FIRST HOP  DRIVER    PCKT_SZ    SUBN_MSK       SUBN_VALUE
162.18         =      MEMD     4096       0.0.255.248      0.0.4.56
DEFAULTNET     =      MEMD     1500       0
!
;START statements
START OFF
!
```

Router Configuration Statements

The following statements configure the offload feature in the router. When you configure a host-to-host communication through the same channel interface, include the **no ip redirects** and **ip route-cache same-interface** commands:

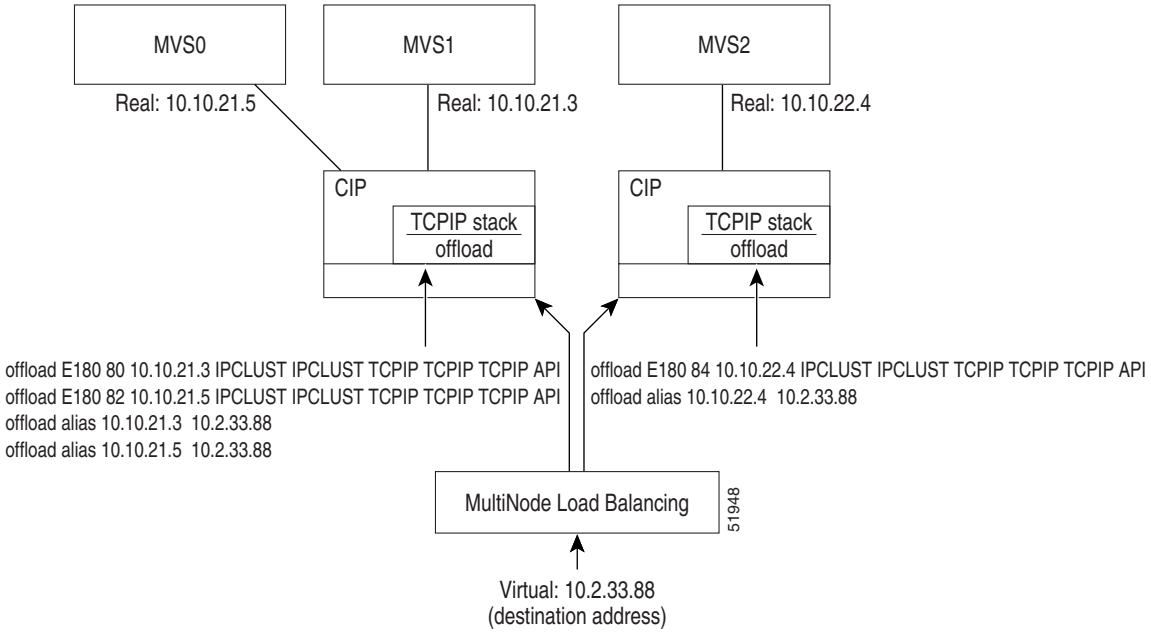
```
interface Channel0/0
 ip address 162.18.4.57 255.255.255.248
 no ip redirects
 ip route-cache same-interface
no keepalive
 offload C300 62 162.18.4.59 CISCOVM CIP1 TCPIP TCPIP TCPIP API
```

Offload Alias Configuration Example

The following example shows an offload alias configuration on two routers with CIP adapters that provide offload support to three MVS hosts.

Figure 251 shows the architecture for this example and the relationship of the IP addresses on the host and offload devices. The alias IP address for each of the MVS hosts is 10.2.33.88 in this example. Each host has a unique real IP address that is associated with the alias IP address in the offload configuration on the CMCC adapter.

Figure 251 Offload Alias Support on Multiple CMCC Adapters to Multiple Hosts



Router 1 Configuration

```
! Select the physical channel interface
!
interface channel 3/1
!
! Assign an IP address to the channel interface
!
 ip address 10.10.21.1 255.255.255.0
!
! Configure other router network characteristics
!
 no ip directed-broadcast
 ip route-cache flow
 no ip mroute-cache
 no ip redirects
 no keepalive
!
! Configure TCP/IP offload and alias support to MVS0 and MVS1
!
offload E180 80 10.10.21.3 IPCLUST IPCLUST TCPIP TCPIP TCPIP API
offload E180 82 10.10.31.5 IPCLUST IPCLUST TCPIP TCPIP TCPIP API
offload alias 10.10.21.3 10.2.33.88
offload alias 10.10.31.5 10.2.33.88
```

Router 2 Configuration

```
! Select the physical channel interface
!
interface channel 3/1
!
! Assign an IP address to the channel interface
!
 ip address 10.10.22.2 255.255.255.0
!
! Configure other router network characteristics
!
 no ip directed-broadcast
 ip route-cache flow
 no ip mroute-cache
 no ip redirects
 no keepalive
!
! Configure TCP/IP offload and alias support to MVS2
!
offload E180 84 10.10.22.4 IPCLUST IPCLUST TCPIP TCPIP TCPIP API
offload alias 10.10.22.4 10.2.33.88
```

IP Host Backup Configuration Example

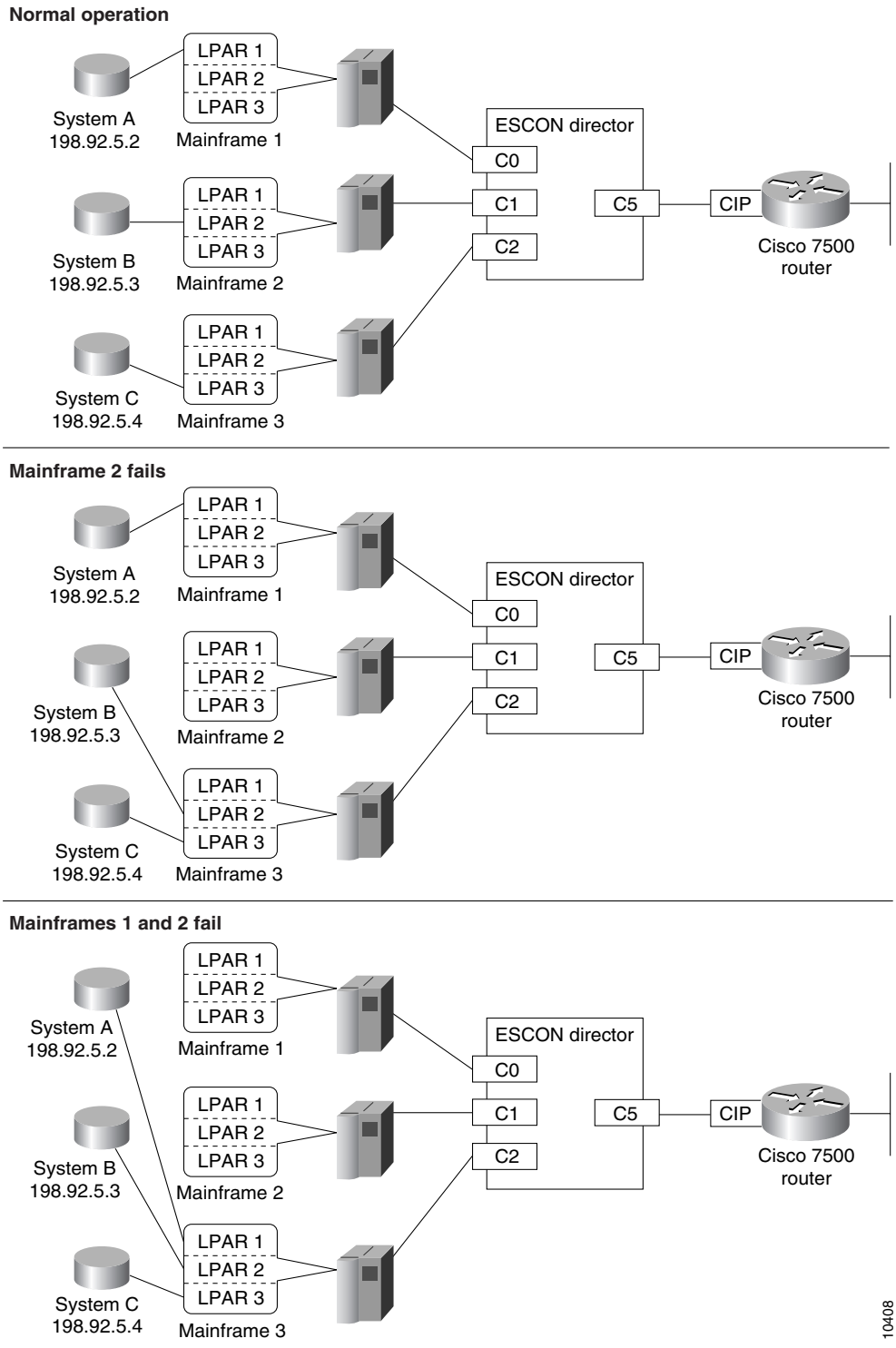
Figure 252 shows the backup connection occurring between System B Mainframe 3 when Mainframe 2 fails.

The intent of this backup configuration is that the system named A will be loaded on one of the mainframes in LPAR1 on that mainframe. The DASD for that system will be shared among all the mainframes but only one of them will ever IPL the system at one time. The same holds for LPAR2 and LPAR3.

The ESCON director has the following connections:

- Cisco 7500 series router with a CIP connected to port C5
- Mainframe1 connected to C0
- Mainframe2 connected to C1
- Mainframe3 connected to C2

Figure 252 IP Host Backup Configuration



The following examples show how to configure IP Host Backup for three mainframe hosts, Mainframe 1, Mainframe 2, and Mainframe 3, as shown in Figure 252. Each mainframe is configured for at least three logical partitions (LPARs).

Excerpts from the host TCP/IP profiles show how the host might be configured. Excerpts from the router configuration show how the IP Host Backup configuration statements are configured.

Host TCP/IP Profiles

The DEVICE and HOME statements in the nine TCP/IP profiles are similar to the following:

```
LPAR1 (mainframes 1, 2, 3):
DEVICE CIP1 CLAW 630 LPAR1 CIP1 NONE 20 20 4096 4096
LINK CIP1L IP 0 CIP1
HOME
 198.92.5.2 CIP1L

LPAR2 (mainframes 1, 2, 3):
DEVICE CIP1 CLAW 730 LPAR1 CIP1 NONE 20 20 4096 4096
LINK CIP1L IP 0 CIP1
HOME
 198.92.5.3 CIP1L

LPAR3 (mainframes 1, 2, 3):
DEVICE CIP1 CLAW 830 LPAR1 CIP1 NONE 20 20 4096 4096
LINK CIP1L IP 0 CIP1
HOME
 198.92.5.4 CIP1L
```

Router Configuration

On the router, the CIP is located in slot 3 and port 1 is connected to the ESCON director. The **path** commands define the group of paths that are used as the IP Host Backup.

```
interface channel 3/1
 ip address 198.92.5.1 255.255.255.128
 path c010 c110 c210
   claw 30 198.92.5.2 lpar1 cip1 tcpip tcpip
 path c020 c120 c220
   claw 30 198.92.5.3 lpar2 cip1 tcpip tcpip
 path c030 c130 c230
   claw 30 198.92.5.4 lpar3 cip1 tcpip tcpip
```

Configuring CSNA and CMPC

Cisco SNA (CSNA) and Cisco Multipath Channel (CMPC) are software features that enable a Cisco router to establish channel connections with a mainframe host. This chapter provides information about configuring the Cisco SNA (CSNA) and Cisco Multipath Channel support on the CIP and CPA types of CMCC adapters on a Cisco router.

This information is described in the following sections:

For a complete description of the CSNA and CMPC commands in this chapter, refer to the "CSNA, CMPC, and CMPC+ Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page lv in the "Using Cisco IOS Software" chapter.

Overview of CSNA and CMPC

This section provides an overview of the architectural and implementation considerations when configuring a CIP or CPA adapter for connection to a mainframe host using the Cisco SNA or Cisco Multipath Channel features. The following topics are included in this section:

- Cisco SNA Environments
- Cisco Multipath Channel Environments

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

Cisco SNA Environments

The CSNA feature provides support for Systems Network Architecture (SNA) protocols to the IBM mainframe from Cisco 7500, Cisco 7200, and Cisco 7000 with RSP7000 series routers, using CMCC adapters (over both ESCON and parallel interfaces). As an IBM 3172 replacement, a CMCC adapter in a Cisco router supports the External Communications Adapter (XCA) feature of the Virtual Telecommunications Access Method (VTAM).

Support for the XCA feature allows Logical Link Control (LLC) downstream physical units (PUs) to be defined as switched devices. XCA support also allows the CMCC adapter to provide an alternative to front-end processors (FEPs) at sites where the Network Control Program (NCP) is not required for SNA routing functions.

The CSNA feature supports communication between a channel-attached mainframe and the following types of devices attached to a LAN or WAN:

- Physical Unit (PU) 2.0 SNA node
- PU 2.1 SNA node
- PU 5/4 SNA node

CSNA also supports communication between two mainframes running VTAM that are either channel-attached to the same CMCC adapter card, or channel-attached to different CMCC adapter cards.

The CSNA feature provides SNA connectivity through a Media Access Control (MAC) address that is defined on an internal adapter in a CMCC. The internal adapter is a virtual adapter that emulates the LAN adapter in an IBM 3172 Interconnect Controller. Each internal adapter is defined in a corresponding XCA major node in VTAM, which provides an access point (LAN gateway) to VTAM for SNA network nodes.

The internal adapter is configured on an internal (virtual) Token Ring LAN located in the CMCC. Each CMCC can be configured with multiple internal Token Ring LANs and internal adapters. Each internal Token Ring LAN must be configured to participate in source-route bridging to communicate with the LAN devices attached to the router.

By providing Cisco Link Services (CLS) and the Logical Link Control type 2 (LLC2) protocol stack on the CMCC adapter card, all frames destined to or from the CMCC adapter card are switched by the router. The presentation of LAN media types allows the CSNA feature to take advantage of current source-route bridging (SRB), remote source-route bridging (RSRB), data-link switching plus (DLSw+), Source-Route Translational Bridging (SR/TLB), internal SDLC-LLC2 translational bridging (SDLLC), Qualified Logical Link Control (QLLC) services, and APPN functionality such as SNA Switching Services (SNASw).

Cisco Multipath Channel Environments

CMPC is Cisco's implementation of IBM's MultiPath Channel (MPC) feature on Cisco 7500, Cisco 7200, and Cisco 7000 with RSP7000 series routers. CMPC allows VTAM to establish Advanced-Peer-to-Peer Networking (APPN) connections using both High Performance Routing (HPR) and Intermediate Session Routing (ISR) through channel-attached router platforms.

Routers configured for CMPC can be deployed in Parallel MVS Systems Complex (sysplex) configurations.

CMPC can be used to establish an APPN connection between VTAM and the following types of APPN nodes:

- VTAM on another host that is channel-attached to the same CMCC adapter
- VTAM on another host that is channel-attached to a different CMCC adapter in the same router
- TN3270 server using Dependent LU Requester (DLUR) in the same CMCC adapter
- SNASw in the router with the CMCC adapter
- Other APPN nodes external to the CMCC adapter and router such as Communications Server/2, AS/400, other LAN- or WAN-attached VTAM hosts, or remote routers

One read subchannel and one write subchannel are supported for each MPC transmission group (TG). The read subchannel and write subchannel may be split over two physical channel connections on the same CMCC adapter.

CMPC insulates VTAM from the actual network topology. The MPC protocols are terminated on the CMCC adapter and converted to LLC protocols. After they are converted to LLC protocols, other Cisco features can be used to connect VTAM to other APPN nodes in the network. CMPC can be used in conjunction with DLSw+, RSRB, SR/TLB, SRB, SDLLC, QLLC, ATM LAN emulation, and FRAS host to provide connectivity to VTAM.

CMPC supports connections to PU 2.1 nodes: APPN NN, APPN EN, and LEN. Subarea connections are not supported.

The CMPC feature can coexist with the CLAW, TCP/IP Offload, IP host backup, CSNA, CMPC+, and TN3270 server features on the same CMCC adapter.

Preparing to Configure CSNA and CMPC

The following topics in this section provide information that is useful when you are planning to configure CSNA or CMPC support:

- Hardware and Software Requirements, page 3
- Mainframe Host Configuration Considerations, page 4

Hardware and Software Requirements

This section provides information about the router and mainframe requirements to support CSNA and CMPC. The router requirements are the same to support either CSNA or CMPC. However, the minimum level of VTAM required on the mainframe varies by whether you are configuring CSNA or CMPC.

Router Requirements

Both the CSNA and CMPC features are supported on the following router platforms:

- Cisco 7500 series—Supports CIP adapters
- Cisco 7200 series—Supports the ECPA and PCPA adapters
- Cisco 7000 series with RSP7000—Supports CIP adapters

You must configure the CSNA and CMPC features on the physical interface of a CMCC adapter. For a CIP, the physical interface is either 0 or 1. For the CPA adapters, ECPA and PCPA, the physical interface is port 0.

Mainframe Requirements

CSNA and CMPC establish channel connectivity to a mainframe host using VTAM on the host. For questions about the required maintenance level or for information about Program Temporary Fixes (PTFs), consult your IBM representative.

The following versions of VTAM are required to configure CSNA and CMPC on a CMCC adapter:

CSNA VTAM Requirement

- VTAM V3.4 and later

CMPC VTAM Requirements

- MPC APPN ISR connections—VTAM V4.2 and later
- MPC APPN HPR connections—VTAM V4.3 and later

Mainframe Host Configuration Considerations

Configuring CSNA or CMPC support requires that you perform tasks for configuration of the mainframe and the router sides of the network environment.

Often in the mixed network environment of mainframes and LANs, an MVS systems programmer installs and maintains the mainframe side of the network, while a network engineer manages the routers on the LAN side of the network. In such an environment, the successful configuration of CSNA or CMPC support requires the close coordination between these job functions at a customer site.

This chapter contains information for both the network engineer and the MVS systems programmer to properly configure the network devices for CSNA or CMPC support. The tasks for configuring CSNA or CMPC support are organized by whether they are host-related configuration tasks or router-related configuration tasks. In addition, a topic for correlating the mainframe and router configuration is provided so that you can identify the dependencies between the host and router configuration elements and be sure that they are set up correctly.

Defining the Channel Subsystem for the Router

To establish the path and allocate the range of subchannel addresses that the CMCC adapter can use for the CSNA or CMPC features, you need to specify the channel subsystem definitions in the Input/Output Control Program (IOCP) or Hardware Configuration Definition (HCD).

For more information about the statements that might be defined in an IOCP file for parallel channels and ESCON channels on the CIP or CPA, see the “Defining the Channel Subsystem for the Router” section in the “Configuring Cisco Mainframe Channel Connection Adapters” chapter of this publication.

Disabling the Missing Interrupt Handler

Because the appropriate configuration of the missing interrupt handler (MIH) varies according to the protocols and software releases used, Cisco offers the following guidance:

- For OS/390 releases Version 2 Release 4 and earlier, set the MIH to zero.
- For OS/390 releases later than Version 2 Release 4 and z/OS releases, refer to the following section of the z/OS Communications Server IP Configuration Reference: http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/f1a1b420/1.2.13?SHELF=f1a1bk31&DT=20020604120755#HDRMOLLY

For information about how to disable the MIH for the unit addresses being used for your CMCC adapter configuration, see the section "Disabling the Missing Interrupt Handler" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter of this publication.

CSNA Support Configuration Task List

CSNA allows CMCC adapters to communicate directly with a mainframe host through VTAM. In this capacity, a CMCC adapter running CSNA can replace the functions of a Token Ring subsystem on a channel-attached front-end processor (FEP) or IBM 3172 Interconnect Controller.

This section describes the configuration tasks required to install CSNA support on the mainframe and router and includes the following topics:

See the "CSNA and CMPC Configuration Examples" section on page 39 for examples.

CSNA Configuration Guidelines

To configure the CSNA feature, you must configure the host VTAM parameters and the CMCC adapter. Consider the following guidelines as you begin to configure CSNA support:

- The CMCC adapters communicate with remote SNA nodes using internal LANs (called virtual or pseudo-rings). An internal LAN can have multiple internal adapters and MAC addresses.
- The CMCC adapters support only the Token Ring type of internal LAN.
- A CMCC adapter can have multiple internal LANs, up to a maximum of 18.

Note Although a CMCC adapter can technically support up to 32 internal LANs, the limit of up to 18 internal adapters on a CMCC adapter makes 18 internal LANs the practical limit.

- A CMCC adapter can have multiple internal adapters, up to a maximum of 18.
- To define the host subchannel (or path) and device, use the **csna** command on the router. The **csna** command is configured on the router's physical channel interface. On a CIP, the physical interface is on ports 0 and 1. On a CPA, the physical interface is always port 0.

- To configure the internal LANs and adapters, use the following ports on a CMCC interface:
  - On a CIP, configure port 2 which is the virtual channel interface.
  - On a CPA, configure port 0 which is the physical channel interface.
- To define the internal LAN adapter used by CSNA on the router, create an XCA major node in VTAM. The XCA major node controls the activation and deactivation of subchannels and SAPs associated with the CMCC internal adapters that are configured for CSNA. One XCA major node is required for each internal LAN adapter to be used by the CSNA feature in the router.
- CSNA can coexist with CLAW, TCP/IP offload, CMPC, CMPC+, and TN3270 server features on the router. When you configure multiple entities on a CMCC adapter, it is important to be sure that you do not introduce SAP conflicts.

  For more information about configuring SAPs, see the "SAP Configuration Guidelines" section in the "Configuring Cisco Mainframe Channel Connections" chapter in this publication.
- CSNA has a limit of 128 SAPs total on the CMCC. So, if you are configuring the TN3270 server using a CSNA connection, the total number of SAPs open on the host plus the number of SAPs defined for PUs on the TN3270 server must be less than or equal to 128.
- If you are configuring CSNA and the TN3270 server on a CMCC, it is good design practice to configure each feature on a separate internal adapter.
- The adapter number that you specify in the **adapter** command on the router must match the adapter number defined in the CSNA XCA major node.
- The host IOCP and HCD parameters must coordinate with the **csna** command parameters on the router and the XCA major node definition to specify the subchannel path, device, and control unit address.
- The unique routing information is determined by a combination of the adapter number, control unit address, and SAP.

CSNA Host Configuration Task List

Configuring CSNA on the mainframe host requires that you establish a path for the CSNA connection by defining the channel subsystem to allocate subchannel addresses, according to the type of router channel connection in use. The tasks in this section assume that the channel subsystem has already been defined to support the CMCC adapter connection.

To establish a SAP for the adapter configured for CSNA in the router, you need to define a VTAM XCA major node. To support the PU type 2.0 and 2.1 connections used in CSNA communication, you need to configure the PU definitions in a switched major node.

This section provides an overview of the primary components needed to implement CSNA on the host. Mainframe systems programmers can use this information as an aid to determine the required parameters to configure CSNA.

The following topics describe the required tasks to configure CSNA on the host:

- Defining the XCA Major Node, page 7
- Defining the Switched Major Node, page 7

Defining the XCA Major Node

To configure the internal LAN adapter that is used for CSNA support on the router and to specify the subchannel and SAP to be used by the host to communicate with the router, you need to define an XCA major node.

To configure the XCA major node for CSNA support in VTAM, you must know the following information:

- A valid subchannel configured in the IOCP or HCD on the host that can be used for CSNA.

  In the following sample configuration, the subchannel address 584 is shown for the CUADDR parameter. In this example, 584 must be one of the available addresses in the IODEVICE statement for the corresponding CMCC channel connection.

- The adapter number configured in the router that identifies the internal LAN adapter. You must define a separate XCA major node for each internal LAN adapter that is configured for CSNA in the router.

  In the following sample configuration, the adapter number 0 is shown for the ADAPNO parameter. In this example, 0 must be the number of the adapter defined on the internal LAN for CSNA use in the CMCC.

VTAM allows SAPs to be defined in multiples of 4. SAP 4 is the most commonly used number for SNA. If you need to define multiple XCA major nodes for multiple internal LAN adapters that are configured for CSNA, you can use the same SAP number of 4 in the XCA major node definition because the ADAPNO parameter uniquely identifies the path.

The following sample configuration shows a sample XCA major node definition (labeled JC27A04) that configures an internal LAN adapter numbered 0 on the router with control unit address 584, and defines a SAP of 4:

```
JC27A04 VBUILD TYPE=XCA
*********************************************************************
PJEC27A PORT ADAPNO=0,                                              X
             CUADDR=584,                                            X
             MEDIUM=RING,                                           X
             SAPADDR=04,                                            X
             TIMER=255
*********************************************************************
JEC27A GROUP DIAL=YES,                                              X
             ANSWER=ON,                                             X
             CALL=INOUT,                                            X
             AUTOGEN=(3,F,E),                                       X
             ISTATUS=ACTIVE
```

Note The primary configuration elements are shown in bold. All parameters followed by a comma in the PORT and GROUP macros require an X in column 72 as a continuation character.

Defining the Switched Major Node

To support Token Ring PU connections to the host through a CMCC adapter in the router, you need to define switched (dial) connections in VTAM in a switched major node. The remote PUs, defined as PU type 2.0 or 2.1 in the VTAM switched major node, represent the remote SNA controllers (such as an IBM 3174). These PUs can include entities such as a PC running 3270 or APPC emulation packages, PUs configured on DSPU, or a TN3270 server.

The following sample configuration shows a sample switched major node definition labeled C0SWN for a CSNA PU:

```
C0SWN VBUILD TYPE=SWNET
C0PU1 PU     ADDR=01,                                                  X
             PUTYPE=2,                                                 X
             IDBLK=05D,                                                X
             IDNUM=C0AA1,                                              X
             MODETAB=ALAMODE,                                          X
             DLOGMODE=SX32702S                                         X
             DISCNT=(NO),                                              X
             USSTAB=USSSNA,                                            X
             ISTATUS=ACTIVE,                                           X
             MAXDATA=521,                                              X
             IRETRY=YES,                                               X
             MAXOUT=7,                                                 X
             PASSLIM=5,                                                X
             MAXPATH=4

C0LU101LU    LOCADDR=02
C0LU102LU    LOCADDR=03
C0LU103LU    LOCADDR=04
C0LU104LU    LOCADDR=05
```

Note The primary configuration elements are shown in bold. All parameters followed by a comma in the PU macro require an X in column 72 as a continuation character.

CSNA Router Configuration Task List

The following sections describe how to configure a CMCC interface for CSNA support. This procedure requires the configuration of both the physical and virtual interfaces on a CIP.

Configuring the CSNA Subchannels

Configuring the CSNA subchannels establishes the physical path between the CMCC interface and the mainframe channel.

To define an SNA subchannel supported by the CSNA feature, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure CSNA. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# csna path device [maxpiu value][time-delay value][length-delay value]`	Defines the CSNA subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. The available options for this command are: • **maxpiu**—(Optional) Maximum packet size (in the range 4096 to 65535 bytes) that the CMCC adapter sends to the host in one I/O operation. The default is 20470 bytes. **Note** Values for a **maxpiu** less than 819 bytes are not recommended because of potential LONGREC errors produced by VTAM. • **time-delay**—(Optional) Maximum allowable delay (in the range 0 to 100 ms) before the CMCC adapter sends packets to the host. The default is 10 ms. • **length-delay**—(Optional) Minimum data length (in the range 0 to 65535 bytes) to accumulate before the CMCC adapter sends packets to the host. The default is 20470 bytes.

Use the **no csna** command to remove the CSNA subchannel device.

Mainframe Configuration Tip

Configuring the subchannel information in the router requires that you correlate the *path* and *device* information from the IOCP or HCD file on the host.

- The *path* argument is a four-digit hexadecimal value that concatenates the path value (2 digits), EMIF partition number (1 digit), and control unit logical address (1 digit).
- The *device* argument is a valid number in the UNITADD range of the IOCP CNTLUNIT statement for the CSNA internal LAN adapter.

For detailed information about how to determine the *path* and *device* values for the **csna** command, see the "Correlating Channel Configuration Parameters" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.

Configuring the Internal LAN

The CSNA feature resides on an internal LAN and adapter in the CMCC on the router. The internal LAN is a virtual Token Ring LAN that is defined within the CIP or CPA on the router. Unlike the CSNA subchannel path that you define on the physical interface of the CMCC, you define the internal LAN on the virtual interface of the CIP. For the CPA, you can only configure the physical interface port.

To configure an internal LAN, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the internal LAN. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# lan tokenring lan-id`	Selects a Token Ring internal LAN interface identified by *lan-id* and enters internal LAN configuration mode. **Note** Token Ring is the only type of internal LAN supported on channel interfaces.

Configuring Internal Adapters

To configure CSNA on the internal LAN, you also need to configure an internal adapter for CSNA use on the LAN. Naming the internal adapter is optional. However, selecting meaningful names for the internal adapters that you configure can simplify identification of the adapter in **show** command output and when troubleshooting is required.

You can configure multiple internal adapters (up to 18) on a CMCC. If you want to support internal adapters with duplicate MAC addresses, you must define the adapter on a different internal LAN and use a unique relative adapter number (RAN). Each internal adapter that is configured for CSNA must have a corresponding XCA major node definition on the host.

To select or configure an internal adapter, use the following commands in internal LAN configuration mode:

	Command	Purpose
Step 1	`Router(cfg-lan)# adapter adapno mac-address`	Selects the internal adapter to configure for CSNA with the following arguments: • *adapno*—Relative adapter number (RAN). • *mac-address*—MAC address for the adapter on the internal LAN. The MAC address cannot be a duplicate on the same internal LAN.
Step 2	`Router(cfg-adap)# name name`	(Optional) Specifies a name for the internal adapter.

Use the **no adapter** command to remove an internal adapter.

Mainframe Configuration Tip

The value for the *adapno* argument in the **adapter** command on the router must match the value specified for the ADAPNO parameter in the corresponding XCA major node definition in VTAM for CSNA. Each internal adapter that is configured for CSNA must have its own XCA major node definition.

Configuring an Internal Adapter's Link Characteristics

To configure the LLC link characteristics of an internal adapter, use the following commands in internal adapter configuration mode, as needed:

Command	Purpose
`Router(cfg-adap)# llc2 n1 bytes`	(Optional) Specifies the maximum size (up to 4105 bytes) of an I-frame. The default is 4105 bytes.
`Router(cfg-adap)# llc2 n2 retry-count`	(Optional) Specifies the maximum retry count (up to 255). The default is 8.
`Router(cfg-adap)# llc2 nw window-size-increase`	(Optional) Increases the window size for consecutive good I-frames received (0 is disabled). The default is 0.
`Router(cfg-adap)# llc2 ack-delay-time milliseconds`	(Optional) Specifies the maximum time (up to 60000 ms) for incoming I-frames to stay unacknowledged. The default is 100 ms.
`Router(cfg-adap)# llc2 ack-max frame-count`	(Optional) Specifies the maximum number of I-frames received (up to 127) before an acknowledgment must be sent. The default is 3.
`Router(cfg-adap)# llc2 idle-time milliseconds`	(Optional) Specifies the frequency of polls (up to 60000 ms) during periods of idle traffic. The default is 60000 ms.
`Router(cfg-adap)# llc2 local-window frame-count`	(Optional) Specifies the maximum number of I-frames to send (up to 127) before waiting for an acknowledgment. The default is 7.
`Router(cfg-adap)# llc2 recv-window frame-count`	(Optional) Controls the number of frames in the receive window. The default is 7.

Command	Purpose
`Router(cfg-adap)# llc2 t1-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for an acknowledgment to send I-frames. The default is 1000 ms.
`Router(cfg-adap)# llc2 tbusy-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) while the other LLC2 station is in a busy state before attempting to poll the remote station. The default is 9600 ms.
`Router(cfg-adap)# llc2 tpf-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for a final response to a poll frame before resending the original poll frame. The default is 1000 ms.
`Router(cfg-adap)# llc2 trej-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for resending a rejected frame before sending the reject command. The default is 3200 ms.

Configuring the Source Bridge

Source-route bridging (SRB) is required to get packets from the LANs that are external to the CMCC adapter, to the internal LAN on the CIP or CPA and the CSNA feature. The **source-bridge** command identifies the interfaces in the same ring group. Frames are sent only to interfaces in the same ring group.

When you configure the source bridge, you can assign the following types of priorities:

- LOCADDR priority—Allows you to maps LUs to queueing priorities for the internal LAN by specifying a defined LOCADDR priority using the **locaddr-priority** command. The LOCADDR priorities are defined using the **locaddr-priority-list** command in global configuration mode.
- SAP priority—Allows you to assign priorities for the internal LAN according to the service access point and MAC address in an LLC2 session by specifying a defined SAP priority using the **sap-priority** command. The SAP priorities are defined using the **sap-priority-list** command in global configuration mode.

To configure the bridging characteristics for the internal LAN, use the following commands in internal LAN configuration mode:

	Command	Purpose
Step 1	`Router(cfg-lan)# source-bridge source-ring-number bridge-number target-ring-number`	Configures source-route bridging for the selected internal LAN interface with the following arguments: • *source-ring-number*—Number for the Token Ring on the internal LAN for the CIP or CPA. • *bridge-number*—Bridge number connecting the source and target Token Rings. • *target-ring-number*—Number of the destination ring number on the router. The target ring can also be a ring group.

	Command	Purpose
Step 2	`Router(cfg-lan)# locaddr-priority list-number`	(Optional) Assigns a LOCADDR priority for the internal LAN, where *list-number* is a value defined from the **locaddr-priority-list** command.
Step 3	`Router(cfg-lan)# sap-priority list-number`	(Optional) Assigns a SAP priority for the internal LAN, where *list-number* is a value defined from the **sap-priority-list** command.

Use the **no source-bridge** command to disable source-route bridging.

Enabling the Router Configuration

After you complete the tasks to configure CSNA on the router, be sure that you enable the configuration using the **no shut** command on all of the applicable interfaces. For the CIP, this means that you need to run the **no shut** command on the selected physical interface, and again for the virtual interface.

For the CPA, you only need to run the **no shut** command on the physical interface.

To enable the router configuration for CSNA, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to 0 or 1 for the physical interface, and 2 for the virtual interface. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# no shut`	Restarts the selected interface.

Correlating the Router and Mainframe Configuration Elements

Table 15 shows a summary of the configuration elements on the router and host that must be correlated for proper operation of CSNA. The column labeled "Configuration Element" identifies the type of entity to be configured. The columns labeled "Router Configuration" and "Mainframe Configuration" identify the related parameters on the router and the mainframe whose values must be compatible or match.

Table 15 Relationship of Router and Mainframe Configuration Elements for CSNA

Configuration Element	Router Configuration	Mainframe Configuration
Subchannels	*path* and *device* arguments of the **csna** command	RESOURCE PARTITION, CHPID, and CNTLUNIT statements of the IOCP definition defining the following parameters for the CSNA channel path: • LPAR number (if defined) in the RESOURCE PARTITION and CHPID statements—Specify in the third digit of the *path* argument in the router **csna** command. • CUADD value (if defined) in the CNTLUNIT statement—Specify in the fourth digit of the *path* argument in the router **csna** command. • Available device address in the UNITADD parameter of the CNTLUNIT statement—Specify in the *device* argument of the router **csna** command.
Internal adapter number	*adapno* argument of the **adapter** command	ADAPNO parameter in the XCA major node definition for the corresponding CSNA internal adapter

CSNA Verification Configuration Task List

Configuring CSNA includes tasks for both the mainframe and the router. This section describes the steps to verify that you have successfully configured CSNA on a CIP. It provides procedures to verify connectivity from the router perspective and from the host perspective, and includes troubleshooting tips as a guide when the configuration verification fails.

This section includes the following topics:

Initial Host and Router Configuration

Consider that you begin the verification procedures with the following sample XCA major node definition, switched major node definition, and initial router configuration:

XCA Major Node Definition

```
JC27A04  VBUILD TYPE=XCA
PJEC27A  PORT ADAPNO=0,                                          X
              CUADDR=27A,                                        X
              SAPADDR=04,                                        X
              MEDIUM=RING,                                       X
              TIMER=244
JEC27A  GROUP ANSWER=ON,                                         X
              AUTOGEN=(3,F,3),                                   X
              CALL=INOUT,                                        X
              DIAL=YES,                                          X
              ISTATUS=ACTIVE
```

Switched Major Node Definition

```
C0SWN VBUILD TYPE=SWNET
C0PU1 PU     ADDR=01,                                             X
             PUTYPE=2,                                            X
             IDBLK=05D,                                           X
             IDNUM=C0AA1,                                         X
             MODETAB=ALAMODE,                                     X
             DLOGMODE=SX32702S                                    X
             DISCNT=(NO),                                         X
             USSTAB=USSSNA,                                       X
             ISTATUS=ACTIVE,                                      X
             MAXDATA=521,                                         X
             IRETRY=YES,                                          X
             MAXOUT=7,                                            X
             PASSLIM=5,                                           X
             MAXPATH=4

C0LU101LU    LOCADDR=02
C0LU102LU    LOCADDR=03
C0LU103LU    LOCADDR=04
C0LU104LU    LOCADDR=05
```

Note The verification procedures assume that the XCA major node and switched major node are defined, but not yet activated.

Router Configuration for Internal LAN on a CIP

```
interface channel 2/1
 no ip address
 no ip directed-broadcast
 no keepalive
!
interface channel 2/2
 no ip redirects
 no ip directed-broadcast
 no keepalive
 lan Token Ring 0
  source-bridge 100 1 400
  adapter 0 4000.8001.0100
```

Note The initial router configuration in the "Router Configuration for Internal LAN on a CIP" section on page 15 shows the internal LAN, source-bridge, and internal adapter configuration in preparation for configuration of CSNA.

Verifying CSNA Channel Connectivity

If you have defined the channel paths for the router at the mainframe host in the IOCP or HCD, you can begin to configure the router for CSNA support and verify connectivity at the channel level first. Isolating this level of verification is useful when the VTAM configuration is not completed, but you want to establish that the router can successfully communicate with the host.

Verifying channel connectivity confirms the following aspects of the router configuration:

- Microcode is loaded on the CMCC
- CMCC adapter is functional
- CMCC can communicate with the host over the channel path

Verifying CSNA Channel Connectivity from the Router

The steps in this section show how to verify the CSNA channel configuration beginning with running the **csna** command on the router's physical interface. The following assumptions are made for the procedure described in this section:

- The router's virtual interface is already configured with the required internal LAN, source-bridge, and internal adapter statements as shown in the initial router configuration for a CIP in the "Router Configuration for Internal LAN on a CIP" section on page 15.
- The router has the recommended CMCC hardware and microcode versions to support the CSNA feature. You can use the **show version**, **show controllers cbus**, and **show controllers channel** commands to verify the Cisco IOS software and CMCC microcode versions.

Note Before you begin on the router, run the **debug channel events** command so that you can verify the messages on the router console.

To verify CSNA channel connectivity, perform the following steps:

Step 1 From the router, configure the **csna** command on the physical interface according to your site's requirements as shown in the following example:

```
interface channel 2/1
csna C190 7A
```

Confirm that you receive a message stating "Device Initialized," similar to the following display:

```
C190-7A Device Initialized
```

Step 2 To verify that the channel is up and the line protocol is up, go to EXEC command mode and run the **show interfaces channel** command as shown in the following example:

```
show interfaces channel 2/1
```

Step 3 To verify that the physical channel is up, run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the path field in the output for the CSNA device shows "ESTABLISHED," which means that the physical channel is up.

Step 4 If your **show** command output matches the values described in Step 2 and Step 3, then the channel connection between the mainframe and the router is established. If you cannot confirm the values, see the "Troubleshooting Tips for Channel Connectivity" section on page 17.

Verifying CSNA Channel Connectivity from the Host

After CSNA has been configured on the router, you can also verify channel connectivity from the host by performing the following steps:

Step 1 From the host, verify that the device is online using the following sample command to display the device:

```
d u,,,27A
```

Step 2 If the device is offline, then vary the device online according to your site's configuration as shown in the following sample command:

```
v 27A,online
```

Note The CHPID for the device should already be active on the host.

Step 3 If the device comes online, then the channel connection between the mainframe and the router is established. If the device does not come online, or you receive the message "No paths physically available," see the "Troubleshooting Tips for Channel Connectivity" section on page 17.

Troubleshooting Tips for Channel Connectivity

There are several indicators on the router and the mainframe that indicate that the channel connection is not available.

- From the router, you might see the following things:
  - The output from the **show interfaces channel** command shows that the channel or line protocol is down.
  - The output from the **show interface channel statistics** command shows that the path is not established (the physical channel is not up).
- From the host, you might see the following things:
  - The device is not online.
  - When you vary the device online, you receive the message "No paths physically available."

Recommended Action

If you determine that the channel connection is not available, review the following tasks to be sure that you have performed them correctly:

- Be sure that you enabled the CSNA router configuration using the **no shut** command to restart the interface. If you configured both the physical and virtual interface on a CIP, be sure to run the **no shut** command on both interfaces.
- Be sure that the CSNA device (and path) are online at the host.
- Verify that the *path* and *device* arguments that you specified in your **csna** configuration command correlate properly to the host IOCP or HCD configuration.

If none of these recommended actions allow you to establish the channel connection, check your CMCC LED indicators and the physical channel connection.

Verifying Communication with VTAM

After the VTAM XCA major node is installed, you can verify communication between the router and VTAM using CSNA. If you have installed a switched major node, you can also verify a session from a network device to the host.

This section includes the following verification procedures:

- Verifying Communication with VTAM from the Host, page 18
- Verifying Communication with VTAM from the Router, page 19
- Troubleshooting Tips for VTAM, page 20

Verifying Communication with VTAM from the Host

This procedure describes how to verify from the host that the XCA major node and switched major node are configured and activated.

To verify communication with VTAM from the host, perform the following steps:

Step 1 If you have configured a switched major node, activate the switched major node from the host using the following sample command:

```
v net,act,id=C0SWN
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I C0PU1 ACTIVE
IST093I C0SWN ACTIVE
```

Step 2 From the host, activate the XCA major node using the following sample command:

```
v net,act,id=JC27A04
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I JC27A04 ACTIVE
IST093I C0SWN ACTIVE
```

If you receive a message similar to the following display, see the "Troubleshooting Tips for VTAM" section on page 20:

```
IST380I ERROR FOR ID=F027A000 - REQUEST: ACTLINK, SENSE: 081C003C
IST380I ERROR FOR ID=F027A001 - REQUEST: ACTLINK, SENSE: 081C003C
IST380I ERROR FOR ID=F027A002 - REQUEST: ACTLINK, SENSE: 081C003C
```

Step 3 (Optional) Using a network station defined with the proper settings, establish a session with the host. In our example, the station should specify the following parameters:

- MAC address of the adapter on the internal LAN as the destination address—4000.8001.0100
- IDBLK/IDNUM (XID) combination for the destination PU, as defined in the switched major node—05DC0AA1
- Destination SAP, as defined in the XCA major node—4

Display the switched major node using the following sample command, and verify that the PU is active and the corresponding LU shows ACT/S:

```
d net, id=C0SWN,e
```

If the PU for the device is not active, see the "Troubleshooting Tips for VTAM" section on page 20.

Verifying Communication with VTAM from the Router

This procedure describes how to verify communication with the VTAM XCA major node for CSNA from the router.

To verify communication with VTAM from the router, perform the following steps:

Step 1 Run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the following is displayed in these fields of the output for the CSNA device:

- Path—The CSNA path is "ESTABLISHED," which means that the physical channel is up.
- Con—The connection value is "Y," which means that the subchannel is up and the CSNA connection is established between the router and the mainframe.

Step 2 To verify that the CMCC adapter has opened a SAP, run the **show extended channel connection-map llc2** command as shown in the following example:

```
show extended channel 2/2 connection-map llc2
```

Step 3 To verify the operational status of the CSNA device, run the **show extended channel csna oper** command as shown in the following example:

```
show extended channel 2/1 csna oper
```

For information about other commands that are useful when diagnosing or monitoring your CSNA connection, see the "Monitoring and Maintaining CSNA and CMPC" section on page 38.

Troubleshooting Tips for VTAM

This section describes recommended actions for the following problems that might occur during verification of communication with VTAM.

- From the router, you might see the following output:
  - The **show interface channel statistics** command shows the field Con=N (the subchannel is not allocated). This output is normal if the XCA major node is not active.
- From the host, you might see the following output:
  - The IST380I message with sense code 081C003C is displayed when you activate the XCA major node.
  - The PU is not active when you display the switched major node after attempting to establish a session.

Recommended Actions

If you encounter problems during verification of communication with VTAM, perform the following tasks to recover:

- If the **show interface channel statistics** command shows that the path is established (the physical channel is up), but the subchannel is not allocated (Con=N), verify that the XCA major node is activated.
- If you receive the sense code 081C003C when activating the XCA major node at the host, review the following tasks to be sure that you have performed them correctly:
  - If you have not already verified channel connectivity, follow the procedure described in the "Verifying CSNA Channel Connectivity" section on page 16.
  - If the channel connectivity is established, verify the configuration values for the adapter number, control unit address, and SAP.

    Be sure that the adapter number that you specified in the XCA major node matches the adapter number on the internal LAN in the router. Verify that the control unit address corresponds to the CSNA device configured on the router and in the IOCP or HCD, and that the SAP is a valid multiple of 4. Be sure that you do not have any SAP conflicts within the router configuration.
- If the PU is not active after attempting to establish a session, verify the values for the following configuration elements in the network device:
  - XID value for the destination device matches the IDBLK/IDNUM value in the switched major node.
  - Destination MAC address matches the MAC address of the internal adapter in the CMCC.
  - Destination SAP address matches the SAP value in the XCA major node. Remember that the SAP address in the XCA major node is in decimal format.

CMPC Support Configuration Task List

CMPC implements the full-duplex IBM channel protocol for SNA, APPN, and HPR traffic. CMPC allows VTAM to establish APPN connections using HPR or ISR through a channel-attached router using a CMCC adapter. CMPC also supports TN3270 using DLUR.

To configure the CMPC feature, you must configure the host VTAM parameters and the CMCC adapter. Consider the following guidelines as you prepare to configure CMPC support:

- The CMCC adapters communicate with remote SNA nodes using internal LANs (called virtual or pseudo-rings). An internal LAN can have multiple internal adapters and MAC addresses.
- The CMCC adapters support only the Token Ring type of internal LAN.
- A CMCC adapter can have multiple internal LANs, up to a maximum of 18.

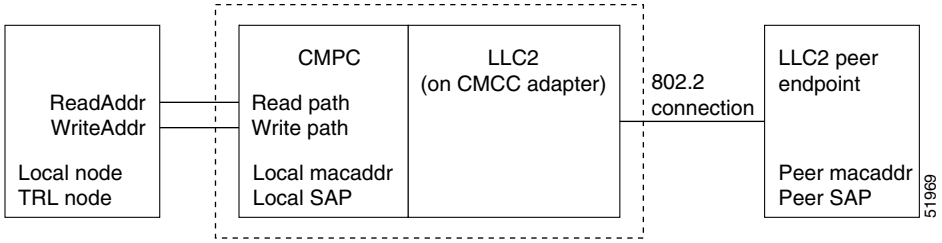
Note Although a CMCC adapter can technically support up to 32 internal LANs, the limit of up to 18 internal adapters on a CMCC adapter makes 18 internal LANs the practical limit.

- A CMCC adapter can have multiple internal adapters, up to a maximum of 18.
- To configure the internal LANs and adapters, use the following ports on a CMCC interface:
  - On a CIP, configure port 2 which is the virtual channel interface.
  - On a CPA, configure port 0 which is the physical channel interface.
- A CMPC link uses two subchannels: one read and one write. Some IBM implementations of MPC allow multiple read and multiple write subchannels within a link. CMPC does not support multiple read and write subchannels. Only one read subchannel and one write subchannel can be configured for each CMPC link. A CMPC link is also referred to as a CMPC TG.
- On the router a CMPC TG consists of one read subchannel definition, one write subchannel definition, and a TG definition, associated by a unique *tg-name*.
- A CMCC adapter can have multiple CMPC links (TGs), up to a maximum of 64.
- To configure the LLC2 interface for the CMCC adapter, use the **tg** (CMPC) command and specify the internal adapter number (which is used to derive the source, or local MAC address) and local SAP address for VTAM. In the **tg** (CMPC) command, you must also identify the remote MAC address and remote SAP of the LLC2 peer with which CMPC communicates. Though this is called the "remote" MAC and SAP, the peer might reside within the router.
- To define the host subchannel (or path) and device, use the **cmpc** command on the router. One **cmpc** command defines the read subchannel, and one **cmpc** command defines the write subchannel. The **cmpc** command is configured on the CMCC adapter's physical interface (port 0 or 1 on a CIP; port 0 on a CPA).
- The two subchannels in a CMPC link do not need to be adjacent devices. Either channel can be the read subchannel or the write subchannel. The two subchannels can be on separate channel process IDs (CHPIDs) in the host.
- The two subchannels must be connected to the same CMCC adapter, however they do not have to be connected to the same physical channel interface on a CIP. On a CIP it is possible to connect a read subchannel to channel interface 0, while the write subchannel is connected to channel interface 1.
- The host IOCP or HCD parameters must coordinate with the **cmpc** command parameters on the router and the transport resource list major node definition to specify the subchannel path, device, and subchannel address.
- To configure MPC on the host, define the Transport Resource List (TRL) and the local SNA major nodes. If you do not plan to support HPR, then you need to disable support in the TRL major node by configuring HPR=NO.

- CMPC can coexist with CLAW, TCP/IP offload, IP host backup, CSNA, CMPC+, and TN3270 server features on the router.
- Only APPN connections are supported across CMPC. For this reason when you configure TN3270 server with CMPC, you must configure the TN3270 server as an APPN end node with DLUR.

As an overview of the configuration process, refer to Figure 253, which shows the CMPC link between the VTAM host, the router, and CMCC adapter card, and the communication to the LLC2 endpoint. The read and write addresses defined in the VTAM host correspond to the read and write paths defined for CMPC. CMPC communicates with the LLC2 stack, which communicates to the endpoint of the connection by means of the IEEE 802.2 link.

Figure 253 Logical View of CMPC Link



This section describes the configuration tasks required to install CMPC support on the mainframe and router and includes the following topics:

- Configuring CMPC on the Host, page 22
- Configuring CMPC on the Router, page 24
- Correlating the Mainframe and Router Configuration Elements, page 29
- CMPC Verification Configuration Task List, page 30

See the "CSNA and CMPC Configuration Examples" section on page 39 for examples.

Configuring CMPC on the Host

Configuring CMPC on the mainframe host requires that you define the TRL and local SNA major nodes. One TRL major node might include several transport resource list entries (TRLEs). The local SNA major node references the TRLE to be used for a specific connection to the control point (CP) in the CMCC.

This section provides an overview of the primary components needed to implement CMPC on the host. Mainframe systems programmers can use this information as an aid to determine the required parameters to configure CMPC.

The following topics describe the required tasks to configure CMPC on the host:

- Configuring the VTAM Transport Resource List Major Node, page 23
- Configuring the VTAM Local SNA Major Node, page 23

Configuring the VTAM Transport Resource List Major Node

To configure MPC on the host, you need to define a Transport Resource List (TRL) major node. To define the TRL, you must have two valid subchannel addresses configured in the IOCP or HCD on the host that can be used for the read and write subchannels. The read/write subchannels that you configure in the TRL should correlate with the unit addresses configured in the *device* argument of the **cmpc** commands.

For details on how to configure the TRL major node, see the following IBM documents:

- *VTAM Resource Definition Samples*, SC31-6554
- *VTAM Operation*, SC31-6549
- *VTAM Network Implementation Guide*, SC31-6548

The following example shows a typical TRL major node configuration:

```
LAGTRLA  VBUILD TYPE=TRL
LAGTRLEA  TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
READ=(2F0),                                                        X
WRITE=(2F1)
```

In this example, device 2F0 has been configured for read and 2F1 has been configured for write. 2F0 and 2F1 must be available subchannels in the IOCP or HCD definition for the CMCC adapter connection.

You should activate the TRL before activating the corresponding local major node. The following example shows the command to activate a TRL, where the ID parameter specifies the name of the TRL, LAGTRLA:

```
v net,act,id=lagtrla,update=add
```

Note that “update=add” is preferred and is the default for later versions of VTAM. The argument “update=all” can cause inactive TRLEs to be deleted unexpectedly from ISTTRL. However, “update=all” must be used if you change an active TRL major node and want the changes to become active.

The following commands are useful for displaying the current list of TRLEs:

- **d net,trl**
- **d net,id=isttrl,e**
- **d net,trl,trle=trle_name**

Configuring the VTAM Local SNA Major Node

To configure the MPC channel link on the VTAM host, define the local SNA major node.

The following is an example of a typical configuration:

```
LAGLNA   VBUILD TYPE=LOCAL
LAGPUA   PU  TRLE=LAGTRLEA,                                        X
             ISTATUS=ACTIVE,                                       X
             XID=YES,CONNTYPE=APPN,CPCP=YES,HPR=YES
```

The TRLE parameter in the local node specifies the label on the TRLE statement from the TRL major node LAGTRLA. If you do not want to run HPR be sure to specify HPR=NO.

Before you activate the local SNA major node, you must activate the TRL node. The following example shows the command to activate a local node, where the ID parameter specifies the name of the local node, LAGLNA:

```
v net,act,id=laglna
```

Configuring CMPC on the Router

The following sections describe how to configure a CMCC interface for CMPC support. This procedure requires the configuration of both the physical and virtual interfaces on a CIP.

Configuring the CMPC Subchannels

Configuring the CMPC subchannels establishes the physical path between the CMCC interface and the mainframe channel.

To define a CMPC read subchannel and CMPC write subchannel, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel` *slot/port*	Selects the interface on which to configure CMPC. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# cmpc` *path device tg-name* `read`	Defines the CMPC read subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. • *tg-name*—Name of the CMPC TG, up to eight characters.
Step 3	`Router(config-if)# cmpc` *path device tg-name* `write`	Defines the CMPC write subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. This unit address must be a different address than the unit address for the CMPC read subchannel. • *tg-name*—Name of the CMPC TG, up to eight characters.

Use the **no cmpc** *path device* command to remove the definition of a subchannel.

Mainframe Configuration Tips

- Configuring the subchannel information in the router requires that you correlate the *path* and *device* information from the IOCP or HCD file on the host.
  - The *path* argument is a four-digit hexadecimal value that concatenates the path value (two digits), EMIF partition number (one digit), and control unit logical address (one digit).
  - The *device* argument is a valid number in the UNITADD range of the IOCP CNTLUNIT statement for the CMPC internal LAN adapter.

  For detailed information about how to determine the *path* and *device* values for the **cmpc** command, see the "Correlating Channel Configuration Parameters" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.
- The **cmpc** commands on the router define the subchannel addresses that CMPC will use to connect to the host, and correspond to the definitions in the TRL major node on the host. Normally, the last two hexadecimal digits in the READ parameter of the TRL match the value of the *device* argument in the corresponding **cmpc read** command. Similarly, the last two hexadecimal digits in the WRITE parameter of the TRL match the value of the *device* argument in the **cmpc write** command.

Configuring the CMPC Transmission Groups

Configuring the CMPC TG defines the MAC/SAP quadruple addressing of an LLC connection. CMPC TGs are configured on the virtual interface of a CIP, and the physical interface of a CPA.

To define a CMPC TG by name and specify its connection to the LLC2 stack, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the CMPC TG. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# tg name llc token-adapter adapter-number lsap [rmac rmac] [rsap rsap]`	Defines the LLC connection parameters for the CMPC TG with the following arguments: • *name*—Name (up to eight characters) of the TG. This name must match the name specified in the **cmpc** command. • *adapter-number*—Relative adapter number of the internal adapter on the CMCC's internal Token Ring LAN. • *lsap*—Local SAP number (multiple of four, from 04 to FC in hexadecimal) to open on the adapter for the connection to VTAM. This SAP number must not conflict with another SAP on the internal adapter for the CMCC. • **rmac** *rmac*—MAC address of a partner link station. • **rsap** *rsap*—SAP address of a partner link station.

The local SAP, remote MAC, and remote SAP parts of the addressing are defined explicitly in the corresponding parameters of the **tg** (CMPC) command. The local MAC address is derived from the internal adapter number that you specify in the *adapter-number* argument. Be sure that you specify a unique local SAP that does not conflict with other SAPs on the same internal adapter.

Use the **no tg** command to remove a CMPC TG from the configuration, which will deactivate the named CMPC TG. To change any parameter of the **tg** statement, the statement must be removed by using the **no tg** *tg-name* command.

Router Configuration Tip

The *name* that you specify for the CMPC TG must match the name that you specify in the *tg-name* argument of the **cmpc** command on the physical interface of the same CMCC adapter.

Configuring the Internal LAN

The CMPC feature resides on an internal LAN and adapter in the CMCC on the router. The internal LAN is a virtual Token Ring LAN that is defined within the CIP or CPA on the router. Unlike the CMPC subchannel path that you define on the physical interface of the CMCC, you define the internal LAN on the virtual interface of the CIP. For the CPA, you can only configure the physical interface port.

To configure an internal LAN, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the internal LAN. The *port* value differs by the type of CMCC adapter: • CIP—*Port* value corresponds to the virtual interface, which is port 2. • CPA—*Port* value corresponds to port 0.
Step 2	`Router(config-if)# lan tokenring lan-id`	Selects a Token Ring internal LAN interface identified by *lan-id* and enters internal LAN configuration mode.

Configuring Internal Adapters

To configure CMPC on the internal LAN, you also need to configure an internal adapter for CMPC use on the LAN. Naming the internal adapter is optional. However, selecting meaningful names for the internal adapters that you configure can simplify identification of the adapter in **show** command output and when troubleshooting is required.

You can configure multiple internal adapters (up to 18) on a CMCC. If you want to support internal adapters with duplicate MAC addresses, you must define the adapter on a different internal LAN and use a unique relative adapter number (RAN).

To select or configure an internal adapter, use the following command in internal LAN configuration mode:

	Command	Purpose
Step 1	`Router(cfg-lan)# adapter adapno mac-address`	Selects the internal adapter to configure for CSNA with the following arguments: • *adapno*—Relative adapter number (RAN). • *mac-address*—MAC address for the adapter on the internal LAN. The MAC address cannot be a duplicate on the same internal LAN.
Step 2	`Router(cfg-adap)# name name`	(Optional) Specifies a name for the internal adapter.

Use the **no adapter** command to remove an internal adapter.

Router Configuration Tip

The value for the *adapno* argument in the **adapter** command on the router must match the value specified in the **tg** (CMPC) command for the CMPC TG.

Configuring an Internal Adapter's Link Characteristics

To configure the LLC link characteristics of an internal adapter, use the following commands in internal adapter configuration mode, as needed:

Command	Purpose
`Router(cfg-adap)# llc2 n1 bytes`	(Optional) Specifies the maximum size (up to 4105 bytes) of an I-frame. The default is 4105 bytes.
`Router(cfg-adap)# llc2 n2 retry-count`	(Optional) Specifies the maximum retry count (up to 255). The default is 8.
`Router(cfg-adap)# llc2 nw window-size-increase`	(Optional) Increases the window size for consecutive good I-frames received (0 is disabled). The default is 0.
`Router(cfg-adap)# llc2 ack-delay-time milliseconds`	(Optional) Specifies the maximum time (up to 60000 ms) for incoming I-frames to stay unacknowledged. The default is 100 ms.
`Router(cfg-adap)# llc2 ack-max frame-count`	(Optional) Specifies the maximum number of I-frames received (up to 127) before an acknowledgment must be sent. The default is 3.
`Router(cfg-adap)# llc2 idle-time milliseconds`	(Optional) Specifies the frequency of polls (up to 60000 ms) during periods of idle traffic. The default is 60000 ms.
`Router(cfg-adap)# llc2 local-window frame-count`	(Optional) Specifies the maximum number of I-frames to send (up to 127) before waiting for an acknowledgment. The default is 7.
`Router(cfg-adap)# llc2 recv-window frame-count`	(Optional) Controls the number of frames in the receive window. The default is 7.
`Router(cfg-adap)# llc2 t1-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for an acknowledgment to send I-frames. The default is 1000 ms.
`Router(cfg-adap)# llc2 tbusy-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) while the other LLC2 station is in a busy state before attempting to poll the remote station. The default is 9600 ms.
`Router(cfg-adap)# llc2 tpf-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for a final response to a poll frame before resending the original poll frame. The default is 1000 ms.
`Router(cfg-adap)# llc2 trej-time milliseconds`	(Optional) Specifies the amount of time to wait (up to 60000 ms) for resending a rejected frame before sending the reject command. The default is 3200 ms.

Configuring the Source Bridge

Source-route bridging (SRB) is required to get packets from the LANs that are external to the CMCC adapter, to the internal LAN on the CIP or CPA and the CMPC feature. The **source-bridge** command identifies the interfaces in the same ring group. Frames are sent only to interfaces in the same ring group.

When you configure the source bridge, you can assign the following types of priorities:

- LOCADDR priority—Allows you to maps LUs to queueing priorities for the internal LAN by specifying a defined LOCADDR priority using the **locaddr-priority** command. The LOCADDR priorities are defined using the **locaddr-priority-list** command in global configuration mode.
- SAP priority—Allows you to assign priorities for the internal LAN according to the service access point and MAC address in an LLC2 session by specifying a defined SAP priority using the **sap-priority** command. The SAP priorities are defined using the **sap-priority-list** command in global configuration mode.

To configure the bridging characteristics for the internal LAN use the following commands in internal LAN configuration mode:

	Command	Purpose
Step 1	`Router(cfg-lan)# source-bridge source-ring-number bridge-number target-ring-number`	Configures source-route bridging for the selected internal LAN interface with the following arguments: • *source-ring-number*—Number for the Token Ring on the internal LAN for the CIP or CPA. • *bridge-number*—Bridge number connecting the source and target Token Rings. • *target-ring-number*—Number of the destination ring number on the router. The target ring can also be a ring group.
Step 2	`Router(cfg-lan)# locaddr-priority list-number`	(Optional) Assigns a LOCADDR priority for the internal LAN, where *list-number* is a value defined from the **locaddr-priority-list** command.
Step 3	`Router(cfg-lan)# sap-priority list-number`	(Optional) Assigns a SAP priority for the internal LAN, where *list-number* is a value defined from the **sap-priority-list** command.

Use the **no source-bridge** command to disable source-route bridging.

Enabling the Router Configuration

After you complete the tasks to configure CMPC on the router, be sure that you enable the configuration using the **no shut** command on all of the applicable interfaces. For the CIP, this means that you need to run the **no shut** command on the selected the physical interface, and again for the virtual interface.

For the CPA, you only need to run the **no shut** command on the physical interface.

To enable the router configuration for CMPC, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to 0 or 1 for the physical interface, and 2 for the virtual interface. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# no shut`	Restarts the selected interface.

Correlating the Mainframe and Router Configuration Elements

Table 16 shows a summary of the configuration elements on the router and host that must be correlated for proper operation of CMPC. The column labeled "Configuration Element" identifies the type of entity to be configured. The columns labeled "Router Configuration" and "Mainframe Configuration" identify the related parameters on the router and the mainframe whose values must be compatible or match.

Table 16 Relationship of Router and Mainframe Configuration Elements for CMPC

Configuration Element	Router Configuration	Mainframe Configuration
Subchannels	*path* and *device* arguments of the **cmpc** command	RESOURCE PARTITION, CHPID, and CNTLUNIT statements of the IOCP definition defining the following parameters for the CMPC channel path: • LPAR number (if defined) in the RESOURCE PARTITION and CHPID statements—Specify in the 3rd digit of the *path* argument in the router **cmpc** command. • CUADD value (if defined) in the CNTLUNIT statement—Specify in the 4th digit of the *path* argument in the router **cmpc** command. • Available device address in the UNITADD parameter of the CNTLUNIT statement—Specify in the *device* argument of the router **cmpc** command.
Read/write subchannels	*device* argument for the **cmpc read** command *device* argument for the **cmpc write** command	Subchannel for the READ parameter of the TRL major node. Subchannel for the WRITE parameter of the TRL major node.

CMPC Verification Configuration Task List

Configuring CMPC includes tasks for both the mainframe and the router. This section describes the steps to verify that you have successfully configured CMPC with the TN3270 server on a CIP. It provides procedures to verify connectivity from the router perspective and from the host perspective, and includes troubleshooting tips as a guide when the configuration verification fails.

This section includes the following topics:

Initial Host and Router Configuration

Consider that you begin verification with the following configurations on the host and router:

Note The verification procedures assume that the VTAM major nodes are defined, but not yet activated.

TRL Major Node Definition

```
JECTRLG VBUILD TYPE=TRL
JCTRLG70 TRLE LNCTL=MPC,                                        X
               MAXBFRU=16,                                      X
               REPLYTO=25.5,                                    X
               MPCLEVEL=NOHPDT,                                 X
               READ=(270),                                      X
               WRITE=(271)                                      X
```

Local SNA Major Node Definition

```
JECLNA VBUILD TYPE=LOCAL
JECPU70 PU TRLE=JCTRLG70,                                        X
           ISTATUS=ACTIVE,                                       X
           XID=YES,                                              X
           CONNTYPE=APPN,                                        X
           CPCP=YES,                                             X
           HPR=YES
```

Switched Major Node Definition

```
SWTNPAN VBUILD TYPE=SWNET,MAXDLUR=4
PANTNPU PU   ADDR=01,                                            X
             PUTYPE=2,                                           X
             IDBLK=415,                                          X
             IDNUM=AAAAA,                                        X
             LUGROUP=DDDJECLU,                                   X
             LUSEED=TNPAN###,                                    X
             ISTATUS=ACTIVE,                                     X
```

```
              MAXDATA=4096,                                          X
              MAXPATH=1
```

LUGROUP Major Node Definition

```
LUJEC  VBUILD TYPE=LUGROUP
DDDJEC  LUGROUP
DYNAMIC LU DLOGMOD=D4C32XX3,                                         X
           MODETAB=ISTINCLM,                                         X
           USSTAB=USSL3270,                                          X
           SSCPFM=USS3270
@       LU DLOGMOD=D4C32784,                                         X
           MODETAB=ISTINCLM,                                         X
           USSTAB=USSL3270,                                          X
           SSCPFM=USS3270
```

Router Configuration for Internal LAN on a CIP with TN3270 Server

```
interface channel 2/1
 no ip address
 no ip directed-broadcast
 no keepalive
!
interface channel 2/2
 ip address 172.18.20.49 255.255.255.248
 no ip redirects
 no ip directed-broadcast
 no keepalive
 lan Token Ring 6
  source-bridge 106 1 400
  adapter 6 4000.8001.0106
 lan Token Ring 7
  source-bridge 107 1 400
  adapter 7 4000.8001.0107
 tn3270-server
  dlur NETA.PANTN32 NETA.MVSG
   lsap token-adapter 6 04
    link HOST2 rmac 4000.8001.0107
   pu PANTNPU 415AAAAA 172.18.20.58
```

Note The initial router configuration shows the internal LAN, source-bridge, and internal adapter configuration in preparation for configuration of CMPC.

Verifying CMPC Channel Connectivity

If you have defined the channel paths for the router at the mainframe host in the IOCP or HCD, you can begin to configure the router for CMPC support and verify connectivity at the channel level first. Isolating this level of verification is useful when the VTAM configuration is not completed, but you want to establish that the router can successfully communicate with the host.

Verifying channel connectivity confirms the following aspects of the router configuration:

- Microcode is loaded on the CMCC
- CMCC adapter is functional
- CMCC can communicate with the host over the channel path

Verifying CMPC Channel Connectivity from the Router

The steps in this section show how to verify the CMPC channel configuration beginning with running the **cmpc** command on the router's physical interface. The following assumptions are made for the procedure described in this section:

- The router's virtual interface is already configured with the required internal LAN, source-bridge, and internal adapter statements as shown in the initial router configuration for a CIP shown in Figure 253.
- The router has the recommended CMCC hardware and microcode versions to support the CMPC feature. You can use the **show version**, **show controllers cbus**, and **show controllers channel** commands to verify the Cisco IOS software and CMCC microcode versions.

Note Before you begin on the router, run the **debug channel events** command so that you can verify the messages on the router console.

To verify CMPC channel connectivity, perform the following steps:

Step 1 From the router, configure the **cmpc** commands on the physical interface according to your site's requirements as shown in the following example:

```
interface channel 2/1
cmpc C190 70 MVSG-TN READ
cmpc C190 71 MVSG-TN WRITE
```

Confirm that you receive messages stating "Device Initialized," similar to the following displays:

```
PA1 MPC C190-70 Device initialized
PA1 MPC C190-71 Device initialized
```

Step 2 Configure the CMPC TG according to your site's requirements as shown in the following example:

```
interface channel 2/2
tg MVSG-TN llc token-adapter 7 04 rmac 4000.8001.0106
```

Confirm that you receive a message stating that the CMPC TG is "Initialized," similar to the following display:

```
CMPC-TG MVSG-TN initialized
```

Step 3 To verify that the channel is up and the line protocol is up, go to EXEC command mode and run the **show interfaces channel** command as shown in the following example:

```
show interfaces channel 2/1
```

Step 4 To verify that the physical channel is up, run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the path field in the output for the CMPC devices shows "ESTABLISHED," which means that the physical channel is up.

Step 5 If your **show** command output matches the values described in Step 3 and Step 4, then the channel connection between the mainframe and the router is established. If you cannot confirm the values, see the "Troubleshooting Tips for Channel Connectivity" section on page 34.

Verifying CMPC Channel Connectivity from the Host

After CMPC has been configured on the router, you can also verify channel connectivity from the host by performing the following steps:

Step 1 From the host, verify that the devices are online using the following sample command to display the device 270 for a range of two (or 270-271):

```
d u,,,270,2
```

Step 2 If the devices are offline, then vary the devices online according to your site's configuration as shown in the following sample commands:

```
v 270,online
v 271,online
```

Note The CHPID for the device should already be active on the host.

Step 3 If the devices come online, then the channel connection between the mainframe and the router is established. If the device does not come online, or you receive the message "No paths physically available," see the "Troubleshooting Tips for Channel Connectivity" section on page 34.

Troubleshooting Tips for Channel Connectivity

There are several indicators on the router and the mainframe that indicate that the channel connection is not available.

- From the router, you might see the following things:
  - The output from the **show interfaces channel** command shows that the channel or line protocol is down.
  - The output from the **show interface channel statistics** command shows that the path is not established (the physical channel is not up).
- From the host, you might see the following things:
  - The device is not online.
  - When you vary the device online, you receive the message "No paths physically available."

Recommended Actions

If you determine that the channel connection is not available, review the following tasks to be sure that you have performed them correctly:

- Be sure that you enabled the CMPC router configuration using the **no shut** command to restart the interface. If you configured both the physical and virtual interface on a CIP, be sure to run the **no shut** command on both interfaces.
- Be sure that the CMPC devices (and paths) are online at the host.
- Verify that the *path* and *device* arguments that you specified in your **cmpc** configuration command correlate properly to the host IOCP or HCD configuration.

If none of these recommended actions allow you to establish the channel connection, check your CMCC LED indicators and the physical channel connection.

Verifying Communication with VTAM

After all of the VTAM major node definitions are installed, you can verify communication between the router and VTAM using CMPC. You can also verify a session from a TN3270 client network device to the host.

This section includes the following verification procedures:

- Verifying Communication with VTAM from the Host, page 35
- Verifying Communication with VTAM from the Router, page 36

Verifying Communication with VTAM from the Host

This procedure describes how to verify from the host that all of the VTAM major node definitions are configured and activated.

To verify communication with VTAM using CMPC, perform the following steps:

Step 1 From the host, activate the switched major node using the following sample command:

```
v net,act,id=SWTNPAN
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I PANTNPU ACTIVE
IST093I SWTNPAN ACTIVE
```

Step 2 Activate the LUGROUP major node using the following sample command:

```
v net,act,id=DDDJEC
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I DDDJEC ACTIVE
```

Step 3 Activate the TRLE using the following sample command:

```
v net,act,id=JCTRLG70,update=add
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I ISTTRL ACTIVE
```

Step 4 Display the TRL status using the command:

```
d net,trl
```

Verify that the TRLE is present but not active, as shown in the following console message:

```
IST1314I TRLE=JCTRLG70 STATUS=INACT CONTROL=MPC
```

Note If the local SNA major node is activated before the TRLE, the TRLE will be active.

Step 5 Activate the local SNA major node using the following sample command:

```
v net,act,id=JCLS270
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I JCLS270 ACTIVE
IEF196I IEF237I O271 ALLOCATED TO TP0271
IEF196I IEF237I 0270 ALLOCATED TO TP0270
IST1086I APPN CONNECTION FOR NETA.PANTN32 IS ACTIVE - TGN = 165
IST093I JECPU70 ACTIVE
IST1096I CP-CP SESSIONS WITH NETA.PANTN32 ACTIVATED
```

Verifying Communication with VTAM from the Router

This procedure describes how to verify communication with the VTAM TRL and local SNA major nodes for CMPC from the router.

To verify communication with VTAM from the router, perform the following steps:

Step 1 Run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the following is displayed in these fields of the output for the CMPC devices:

- Path—The CMPC path is "ESTABLISHED," which means that the physical channel is up.
- Con—The connection value is "Y," which means that the subchannel is up and the CMPC connection is established between the router and the mainframe.

Step 2 To verify that the CMPC subchannels are active, run the **show extended channel cmpc** command as shown in the following example:

```
show extended channel 2/0 cmpc
```

Step 3 To verify the operational status and configuration of the CMPC TGs, run the **show extended channel tg** command as shown in the following example:

```
show extended channel 2/2 tg detailed MVSG-TN
```

For information about other commands that are useful when diagnosing or monitoring your CMPC connection, see the "Monitoring and Maintaining CSNA and CMPC" section on page 38.

Troubleshooting Tips for VTAM

This section describes recommended actions for the following problems that might occur during verification of communication with VTAM.

- When you activate the local SNA major node, you receive the following messages:

```
IST259I INOP RECEIVED FOR JECPU70 CODE=01
IST6191 ID = JECPU70 FAILED - RECOVERY IN PROGRESS
IST129I UNRECOVERABLE OR FORCED ERROR ON NODE JECPU70 - VARY INACT SCHED
IST105I JECPU70 NODE NOW INACTIVE
```

Recommended Actions

  - Be sure that the CMPC devices (and paths) are online at the host.
  - Verify that the *path* and *device* arguments that you specified in your **cmpc** configuration commands correlate properly to the host IOCP or HCD configuration and to the TRL major node.

- The local SNA major node activates and the subchannels are allocated, but you receive a message similar to the following display on the router console:

```
MPC-6-NODE_NOT_ACTIVE: Host attempted activation of MVSG-TN but TG not configured
```

Recommended Actions

  - Verify that the TG is defined on the router.
  - Verify that you specified the same TG name in the **tg** (CMPC) command and in each of the **cmpc** commands.

- When you activate the local SNA major node, you receive the following messages at the host:

```
IST097I VARY ACCEPTED
IST093I JCLS270 ACTIVE
IEF196I IEF237I O271 ALLOCATED TO TP0271
IEF196I IEF237I 0270 ALLOCATED TO TP0270
IST222I READ DEVICE 0271 IS INOPERATIVE, NAME ISJCTRLG70 446
IST1578I DEVICE INOP DETECTED FOR JCTRLG70 BY ISTTSCXI CODE = 200
IST314I END
IST1222I WRITE DEVICE 0270 IS INOPERATIVE, NHAME IS JCTRLG70 447
IST1578I DEVICE INOP DETECTED FOR JCTRLG70 BY ISTTSCXI CODE = 200
IST314I END
IST1578I SOFT INOP DETECTED FOR JCTRLG70 BY ISTTSC8X CODE = 007
IST259I INOP RECEIVED FOR JECPU70 CODE = 01
IST619I ID = JECPU70 FAILED - RECOVERY IN PROGRESS
IST129I UNRECOVERABLE OR FORCED ERROR ON NODE JECPU70 - VARY INACT SCHED
IST105I JECPU70 NODE NOW INACTIVE
```

  In addition, you receive messages similar to the following display on the router console:

```
MPC-6-BAD_DIRECTION:PA1 MPC C190-70 configured for READ
MPC-6-BAD_DIRECTION:PA1 MPC C190-71 configured for WRITE
```

Recommended Action

Verify that the direction (read versus write) that you specified for the subchannel in the TRLE matches the direction that you specified in the **cmpc** commands. So, the host READ subchannel matches the **cmpc read** device and the host WRITE subchannel matches the **cmpc write** device.

Monitoring and Maintaining CSNA and CMPC

The following topics in this section provide information about the different commands that you can use to monitor and maintain the CMCC interfaces that are configured for CSNA and CMPC:

Monitoring Interface Status

To monitor CMCC adapter interface status, you can display information about the interface, including the version of the software and the hardware, the controller status, and statistics about the interfaces. In addition, you can display information about feature-related statistics on the CMCC adapter. This section lists some additional commands that are useful when monitoring CMCC adapter interfaces that are configured for CSNA and CMPC.

For a complete list of the **show** commands that are related to monitoring CMCC adapter interfaces, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication. To display the full list of **show** commands, enter **show ?** at the EXEC prompt.

To display information related to CSNA and CMPC configurations, use the following commands in EXEC mode:

Command	Purpose
Router# **show extended channel** *slot/port* **csna** [**admin** | **oper** | **stats**] [*path* [*device*]]	Displays information about the CSNA subchannels configured on the specified CMCC adapter interface.
Router# **show extended channel** *slot/port* **cmpc** [*path* [*device*]]	Displays information about each CMPC (and CMPC+) subchannel configured on the specified CMCC adapter interface.
Router# **show extended channel** *slot/port* **tg** [**oper** | **stats**] [**detailed**] [*tg-name*]	Displays configuration, operational, and statistics information for CMPC (and CMPC+) TGs configured on a specified CMCC adapter internal LAN interface.
Router# **show extended channel** *slot/port* **connection-map llc2**	Displays the number of active LLC2 connections for each SAP and the mapping of the internal MAC adapter and the SAP to the resource that activated the SAP.
Router# **show extended channel** *slot/port* **llc2** [**admin** | **oper** | **stats**] [*lmac* [*lsap* [*rmac* [*rsap*]]]]	Displays information about the LLC2 sessions running on the CMCC adapter interfaces.
Router# **show extended channel** *slot/port* **max-llc2-sessions**	Displays information about the number of LLC2 sessions supported on the CMCC adapter.

Clearing Counters for CSNA and CMPC

You can reset the statistics counters that are displayed in the output of the **show extended channel** commands. You can reset the counters associated with an interface or a particular feature on the interface. If you are monitoring a particular threshold or statistic for CSNA or CMPC and need to reset a related counter, you can clear all those counters related to the feature.

For information about clearing other counters on the CMCC adapter interface, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.

To clear the counters associated with CSNA and CMPC on the CMCC adapters, use the following commands in privileged EXEC mode:

Command	Purpose
`router# clear extended counters channel slot/port csna`	Clears counters for statistics associated with the CSNA feature on the specified *slot/port*. The port value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to the physical interface, which is port 0.
`router# clear extended counters channel slot/port tg`	Clears counters for statistics associated with TGs in the CMPC (and CMPC+) features on the specified *slot/port*. The port value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to the physical interface, which is port 0.
`router# clear extended counters channel slot/port llc2`	Clears counters for LLC2 statistics on the specified *slot/port*. The port value differs by the type of CMCC adapter. • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to the physical interface, which is port 0.

Note These commands will not clear counters retrieved using Simple Network Management Protocol (SNMP), but only those seen with the EXEC **show extended channel** commands.

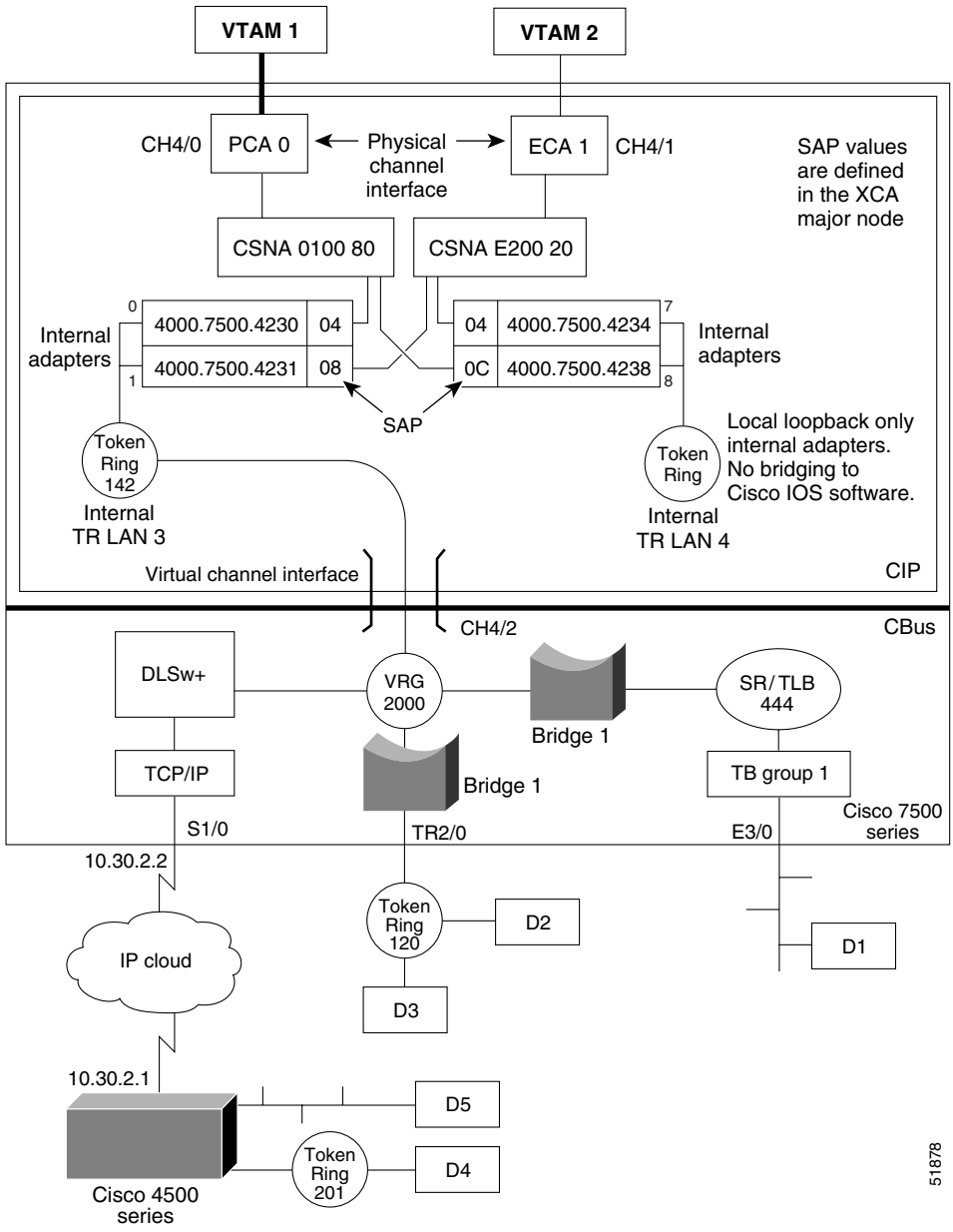
CSNA and CMPC Configuration Examples

The configuration examples in this section are organized by the following categories:

CSNA Configuration on a CIP Example

Figure 254 illustrates an example of configuring CSNA on a Cisco 7500 router with a CIP.

Figure 254 CIP CSNA Source-Route Translational Bridging Configuration



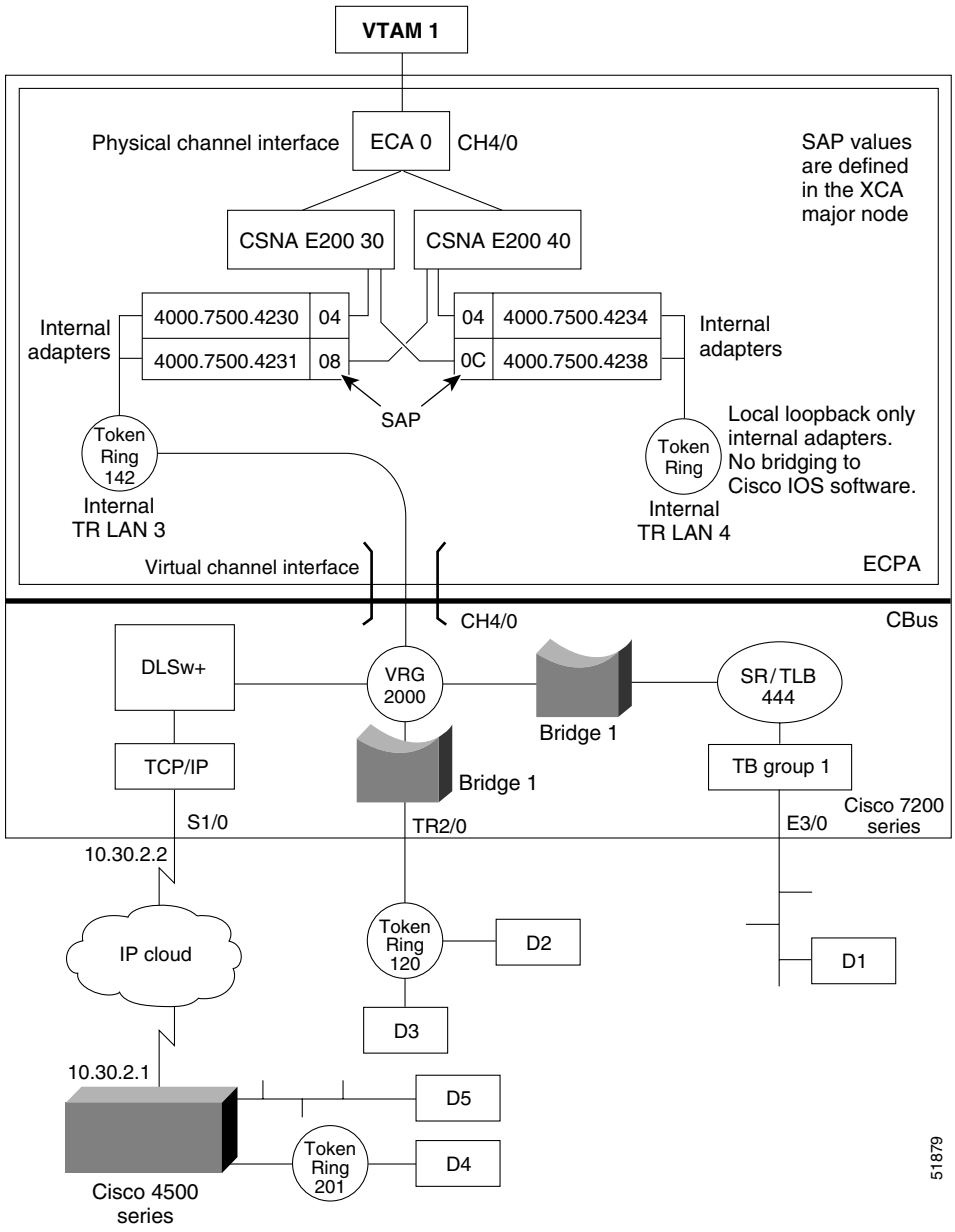

```
source-bridge ring-group 2000
source-bridge transparent 2000 444 1 1
dlsw remote-peer 0 tcp 10.30.2.1
dlsw local-peer peer-id 10.30.2.2
!
interface serial 1/0
 ip address 10.30.2.2 255.255.255.128
 clockrate 56000
!
interface tokenring 2/0
 mac-address 4000.7500.0200
 no ip address
 ring-speed 16
 source-bridge 120 1 2000
 source-bridge spanning
```

```
!
interface ethernet 3/0
 mac-address 0200.ae00.c000
 no ip address
 bridge-group 1
!
interface channel 4/0
 no ip address
 no ip directed-broadcast
 no keepalive
 csna 0100 80
!
interface channel 4/1
 no ip address
 no ip directed-broadcast
 no keepalive
 csna E200 20 maxpiu 65535 time-delay 100
!
interface channel 4/2
 no ip address
 no ip directed-broadcast
 no keepalive
 max-llc2-sessions 2500
 lan TokenRing 3
  source-bridge 142 1 2000
  adapter 0 4000.7500.4230
   llc2 local-window 1
   llc2 ack-max 1
  adapter 1 4000.7500.4231
 lan TokenRing 4
  adapter 7 4000.7500.4234
  adapter 8 4000.7500.4238
!
bridge 1 protocol ieee
```

CSNA Configuration on an ECPA Example

The following configuration is an example of configuring CSNA on a Cisco 7200 router with a ECPA. Figure 255 illustrates this configuration example.

Figure 255 ECPA CSNA Source-Route Translational Bridging Configuration




```
source-bridge ring-group 2000
source-bridge transparent 2000 444 1 1
dlsw remote-peer 0 tcp 10.30.3.1
dlsw local-peer peer-id 10.30.2.2
!
interface serial 1/0
 ip address 10.30.2.2 255.255.255.128
 clockrate 56000
!
interface tokenring 2/0
 mac-address 4000.7500.0200
 no ip address
 ring-speed 16
 source-bridge 120 1 2000
 source-bridge spanning
```

```
!
interface ethernet 3/0
 mac-address 0200.ae00.c000
 no ip address
 bridge-group 1
!
interface channel 4/0
 no ip address
 no ip directed-broadcast
 no keepalive
 csna E200 30 maxpiu 65535
 csna E200 40 maxpiu 65535
 max-llc2-sessions 2500
 lan TokenRing 3
  source-bridge 142 1 2000
  adapter 0 4000.7500.4230
   llc2 local-window 1
   llc2 ack-max 1
  adapter 1 4000.7500.4231
 lan TokenRing 4
  adapter 7 4000.7500.4234
  adapter 8 4000.7500.4238
!
bridge 1 protocol ieee
```

CMPC Configuration Examples

This section provides sample configurations for the CMPC feature. Throughout these configuration samples, a Cisco 7500 router with an RSP is used to illustrate the configurations. The configurations also apply to a Cisco 7000 router with an RP or an RSP installed. All SAP values are written in hexadecimal form.

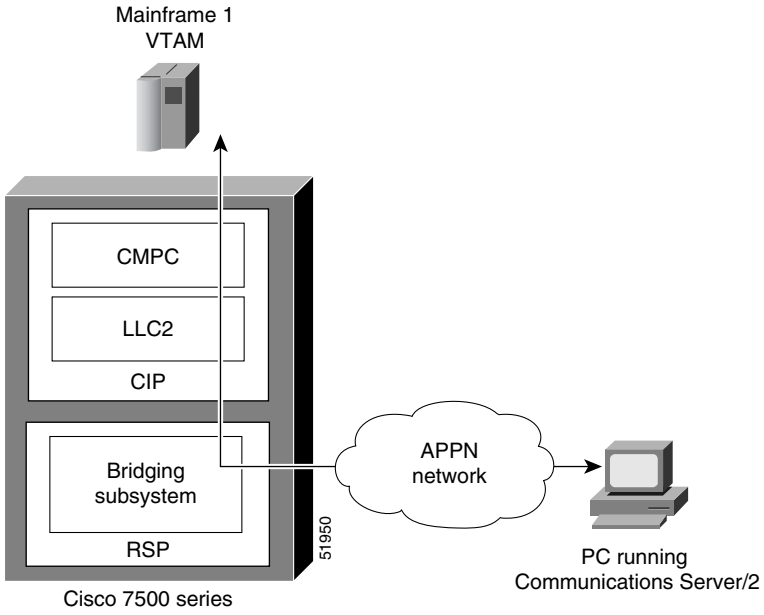
This sections includes the following configuration examples:

- Connecting VTAM to a Remote PC with Communications Server/2 Using CMPC Example, page 44
- Connecting VTAM to SNA Switching Services (SNASw) on the RSP Using CMPC Example, page 46
- Connecting Two VTAM Nodes Using Two CIPs in the Same Router and CMPC Example, page 49
- Connecting VTAM to SNASw on a Remote Router with DLUR Using CMPC Example, page 52

Connecting VTAM to a Remote PC with Communications Server/2 Using CMPC Example

Figure 256 shows the physical components for this example. Figure 257 shows the various parameters for each component in the configuration example.

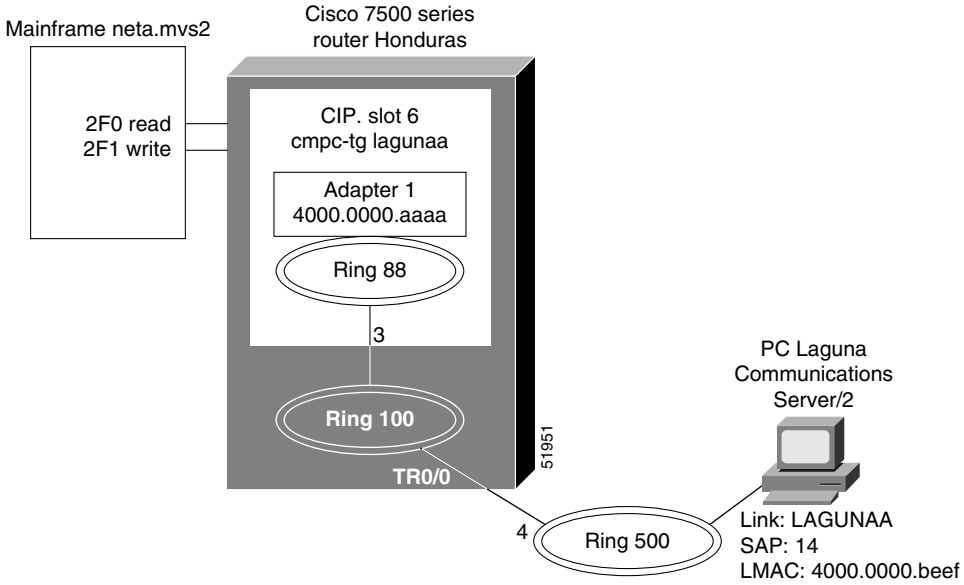
Figure 256 Topology for VTAM-to-Remote PC with Communications Server/2



In Figure 256, the following activity occurs:

- VTAM connects to the CMPC driver on the CIP.
- The CMPC driver converts the data to an LLC data stream and passes the data to the LLC2 stack on the CIP.
- The LLC2 stack on the CIP passes the data to the bridging code on the RSP.
- The bridging code on the RSP passes the data to the APPN network.

Figure 257 Parameters for VTAM-to-Remote PC with Communications Server/2



The example in Figure 257 shows CMPC running on the CIP and communicating with a PC running Communications Server/2. APPN is not running on the router. It is only running in VTAM and on the PC.

The configuration examples for the VTAM host and the router follow.

TRL Node LAGTRLA on MVS2

```
LAGTRA   VBUILD TYPE=TRL
LAGTRLA  TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
               READ=(2F0),                                        X
               WRITE=(2F1)
```

Local Node LAGLNA on MVS2

```
LAGLNA   VBUILD TYPE=LOCAL
LAGPUA   PU    TRLE=LAGTRLA,                                      X
               ISTATUS=ACTIVE,                                    X
               XID=YES,CONNTYPE=APPN,CPCP=YES,HPR=YES
```

Configuration for Honduras Router

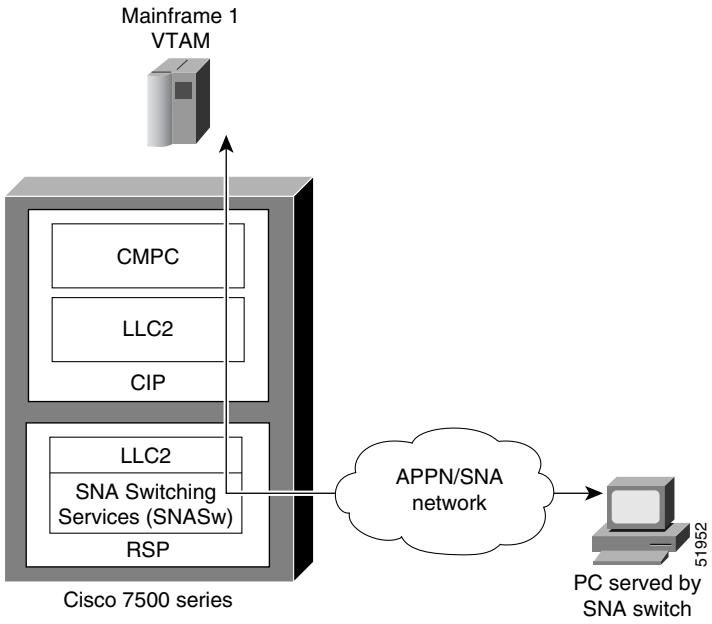
```
source-bridge ring-group 100
!
interface TokenRing0/0
 no ip address
 ring-speed 16
 source-bridge 500 4 100
!
interface Ethernet1/0
 ip address 172.18.3.24 255.255.255.0
!
interface Channel6/1
 no ip address
 no keepalive
 cmpc C020 F0 LAGUNAA READ
 cmpc C020 F1 LAGUNAA WRITE
!
```

```
interface Channel6/2
 no ip address
 no keepalive
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 1 4000.aaaa.aaaa
 tg LAGUNAA  llc token-adapter 1  18 rmac 4000.0000.beef rsap 14
```

Connecting VTAM to SNA Switching Services (SNASw) on the RSP Using CMPC Example

Figure 258 shows the physical components for this example. Figure 259 shows the various parameters for each component in the configuration example.

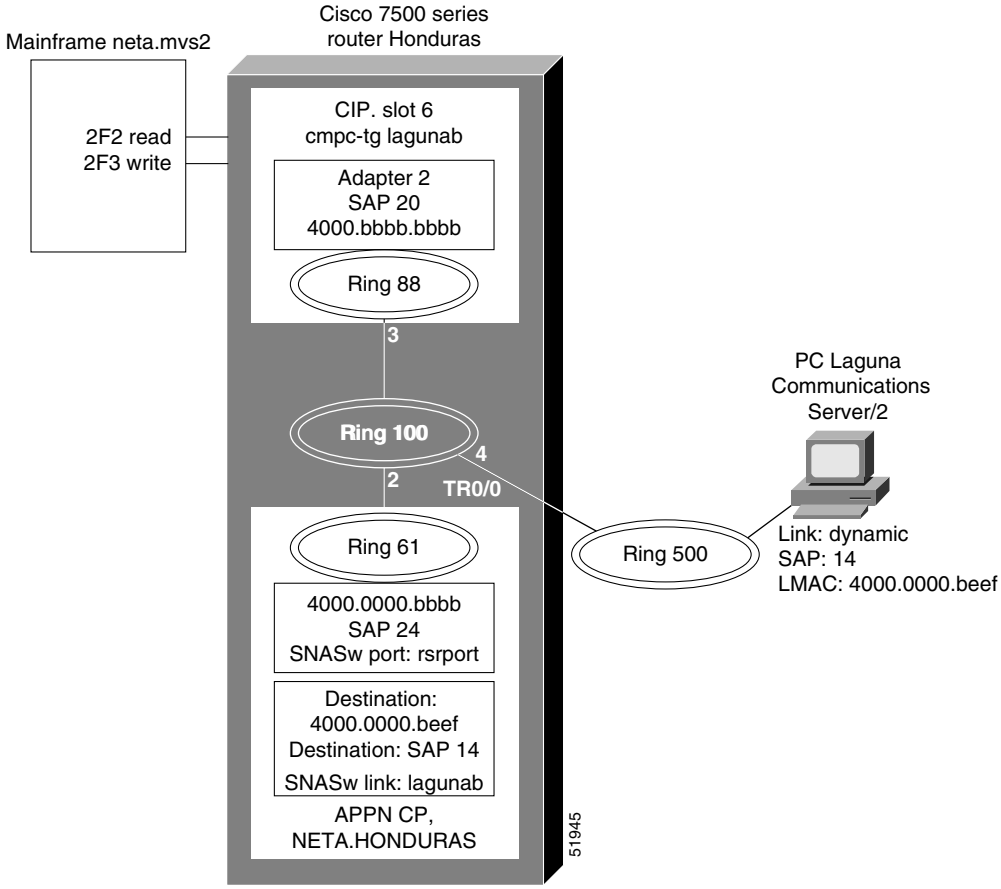
Figure 258 Topology for VTAM-to-SNASw Connection on the CIP



In Figure 259, the following activity occurs:

- VTAM connects to the CMPC driver on the CIP.
- The CMPC driver converts the data to an LLC data stream and passes the data to the LLC2 stack on the CIP.
- The LLC2 stack on the CIP passes the data to the LLC2 stack on the RSP.
- The LLC2 stack on the RSP passes the data to SNA Switching Services on the RSP.
- SNA Switching Services on the RSP sends the data to the APPN/SNA network.

Figure 259 Parameters for VTAM-to-SNASw Connection on the CIP



The configuration illustrated in Figure 259 is more complex because you must configure SNASw on the router. There are many different ways to configure SNASw. The example is a simple SNASw configuration in which SRB is used to connect the SNASw on the RSP to VTAM and the Token Ring attached PC.

It is possible to connect directly to the Token Ring port, which is not shown in the example.

Configuration for TRL Node LAGTRLB

```
LAGTRB   VBUILD TYPE=TRL
LAGTRLB  TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
               READ=(2F2),                                        X
               WRITE=(2F3)
```

Local SNA Major Node LAGLNB

```
LAGNNB   VBUILD TYPE=LOCAL
LAGPUB   PU    TRLE=LAGTRLB,                                       X
               ISTATUS=ACTIVE,                                    X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

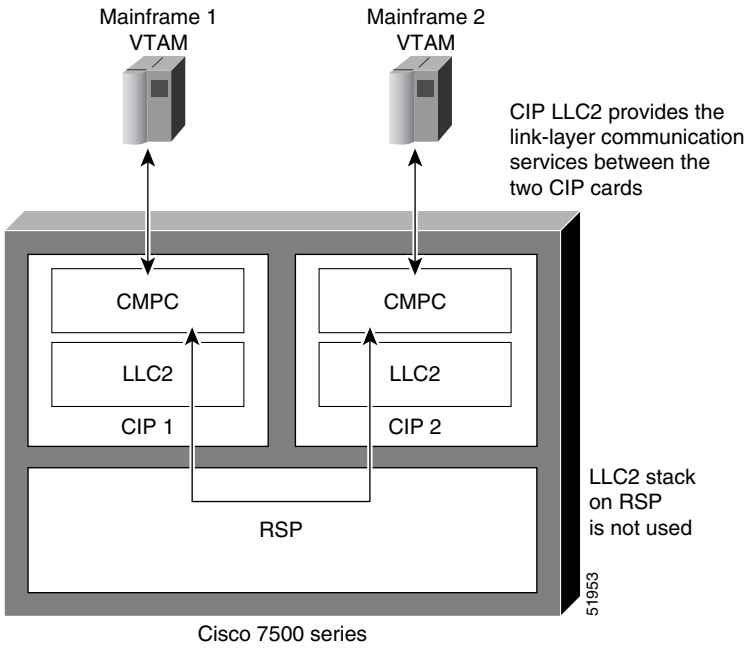
Honduras Router

```
source-bridge ring-group 100
!
interface Channel6/1
 no ip address
 no keepalive
 cmpc C020 F2 LAGUNAB READ
 cmpc C020 F3 LAGUNAB WRITE
!
interface Channel6/2
 no ip address
 no keepalive
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 2 4000.bbbb.bbbb
 lan TokenRing 2
 tg LAGUNAB  llc token-adapter 2  20 rmac 4000.0000.bbbb rsap 24
!
!
interface Virtual-TokenRing0
 mac-address 4000.0000.bbbb
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 61 2 100
!
snasw cpname NETA.HONDURAS
snasw port VTOK Virtual-TokenRing0
snasw link MVS2D port VTOK rmac 4000.bbbb.bbbb
```

Connecting Two VTAM Nodes Using Two CIPs in the Same Router and CMPC Example

Figure 260 shows the physical components for this example. Figure 261 shows the various parameters for each component in the configuration example.

Figure 260 Topology for VTAM-to-VTAM Connection



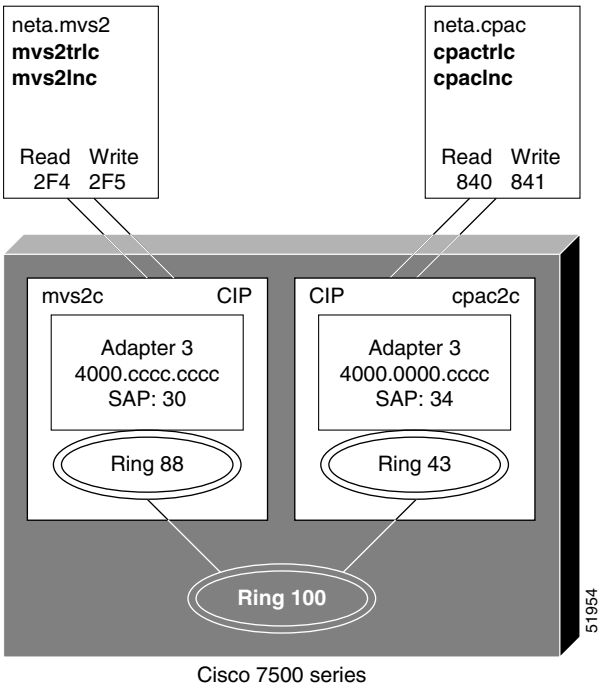
In Figure 260, the following activity occurs:

- VTAM on Mainframe 1 passes MPC data to the CMPC driver on CIP 1.
- The CMPC driver on CIP 1 passes the data to the LLC2 stack.
- LLC2 sends the data to CIP 2 in the same router via IEEE 802.2.
- The LLC2 stack on CIP 2 passes the data to the CMPC driver on CIP 2, which passes the data to VTAM on Mainframe 2.

The CIPs could be in different routers or both VTAM connections could be to the same CIP.

Figure 261 shows parameters for VTAM-to-VTAM connection.

Figure 261 Parameters for VTAM-to-VTAM Connection



Differing solutions can be configured for the example shown in Figure 261. For example, you can have two CIPs in different routers connected via LLC2. You can also configure host connections going into the same CIP card rather than two separate CIP cards.

mvs2trlc

```
MVS2TRC  VBUILD TYPE=TRL
MVS2TRLC TRLE   LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
                READ=(2F4),                                         X
                WRITE=(2F5)
```

mvs2lnc

```
MVS2NNC  VBUILD TYPE=LOCAL
MVS2PUC  PU     TRLE=MVS2TRLC,                                      X
                ISTATUS=ACTIVE,                                     X
                XID=YES,CONNTYPE=APPN,CPCP=YES
```

cpactrlc

```
CPACTRC  VBUILD TYPE=TRL
CPACTRLC TRLE   LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
                READ=(840),                                         X
                WRITE=(841)
```

cpaclnc

```
CPACNNC  VBUILD TYPE=LOCAL
CPACPUC  PU    TRLE=CPACTRLC,                                          X
               ISTATUS=ACTIVE,                                         X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

Router

```
source-bridge ring-group 100
!
interface Channel4/1
 no ip address
 no keepalive
 cmpc C010 40 CPACC READ
 cmpc C010 41 CPACC WRITE
!
interface Channel4/2
 no ip address
 no keepalive
 lan TokenRing 0
  source-bridge 43 5 100
  adapter 3 4000.0000.cccc
 tg CPACC    llc token-adapter 3  34 rmac 4000.cccc.cccc rsap 30
!
interface Channel6/1
 no ip address
 no keepalive
 cmpc C020 F4 MVS2C READ
 cmpc C020 F5 MVS2C WRITE
!
interface Channel6/2
  lan TokenRing 0
  source-bridge 88 3 100
    adapter 3 4000.cccc.cccc
 tg MVS2C    llc token-adapter 3  30 rmac 4000.0000.cccc rsap 34
```

Connecting VTAM to SNASw on a Remote Router with DLUR Using CMPC Example

Figure 262 shows the physical components for the DLUS-to-DLUR configuration. Figure 263 shows the various parameters for each component in the configuration example.

Figure 262 Topology for VTAM-to-SNASw on a Remote Router with DLUR Connection

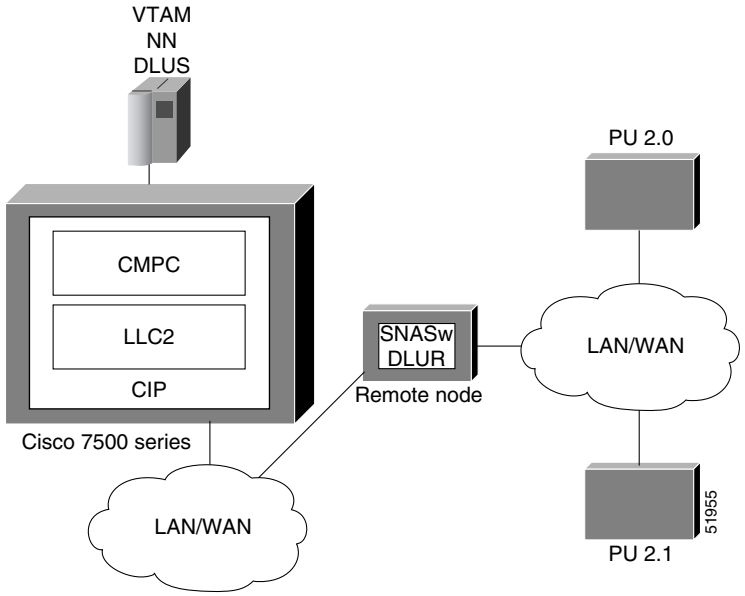
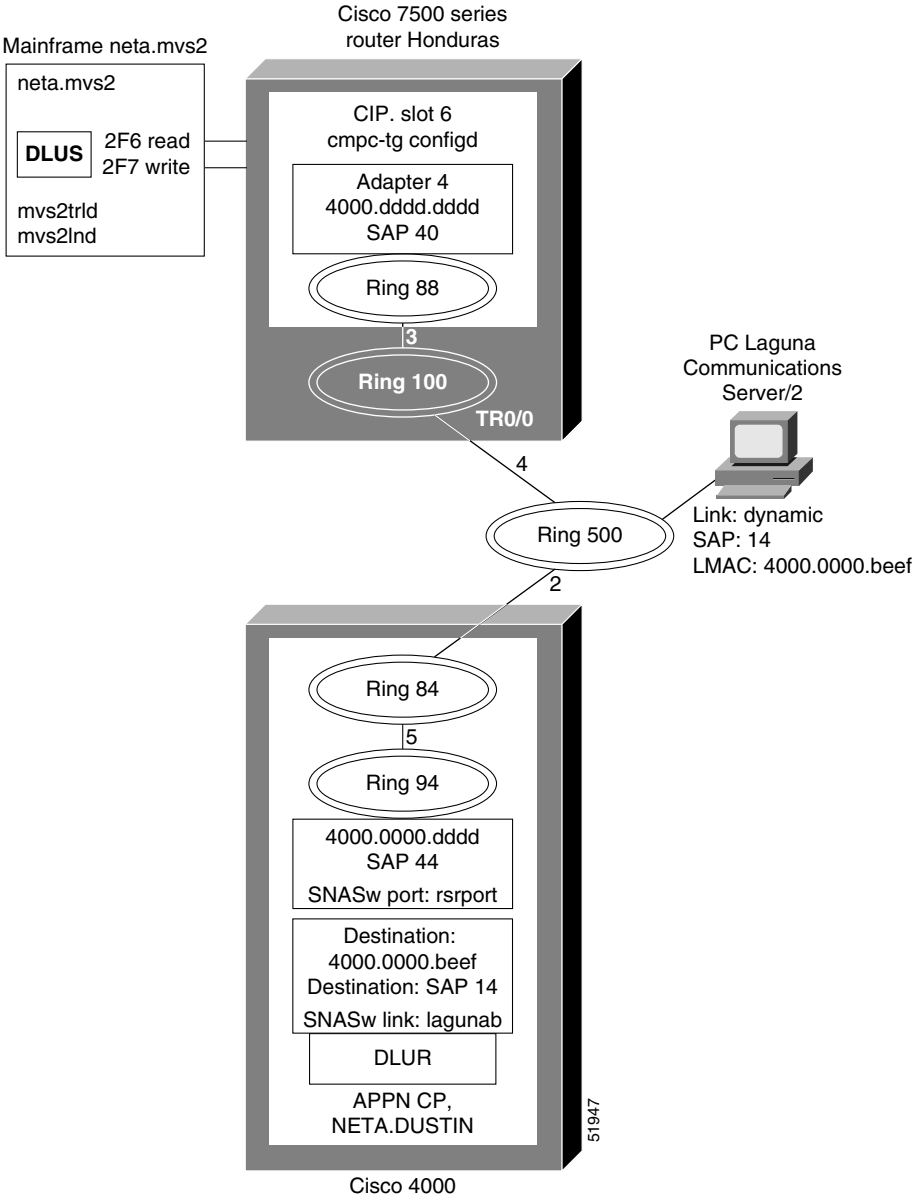


Figure 263 Parameters for VTAM-to-SNASw on a Remote Router with DLUR Connection



In the example shown in Figure 263, DLUS is running on the MVS host. DLUR is running on a remote Cisco 4000 router. The connection from MPC to the APPN stack on the Cisco 4000 is via LLC2. There is no NN on the Cisco 7500. The PC is running Communications Server/2.

mvs2trld

```
MVS2TRD  VBUILD TYPE=TRL
MVS2TRLD TRLE   LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,                    X
                READ=(2F6),                                         X
                WRITE=(2F7)
```

mvs2lnd

```
MVS2NND  VBUILD TYPE=LOCAL
MVS2PUD  PU    TRLE=MVS2TRLD,                                          X
               ISTATUS=ACTIVE,                                         X
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

Additional Configuration for Router Honduras

```
interface Channel6/1
 cmpc C020 F6 CONFIGD WRITE
 cmpc C020 F7 CONFIGD READ
!
interface Channel6/2
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 4 4000.dddd.dddd
  tg CONFIGD  llc token-adapter 4  40 rmac 4000.0000.dddd rsap 44
```

Router Dustin

```
source-bridge ring-group 84
interface Ethernet0
 ip address 172.18.3.36 255.255.255.0
 media-type 10BaseT
!
interface TokenRing0
 no ip address
 ring-speed 16
 source-bridge 500 2 84
!
interface Virtual-TokenRing0
 mac-address 4000.0000.dddd
 no ip address
 no ip directed-broadcast
 ring-speed 16
 source-bridge 94 5 84
!
snasw cpname NETA.DUSTIN
snasw port VTOK Virtual-TokenRing0
snasw link MVS2D port VTOK rmac 4000.dddd.dddd
```

Configuring CMPC+

Cisco Multipath Channel+ (CMPC+) is Cisco's implementation of IBM's Multipath Channel+ (MPC+) feature. The CMPC+ feature in Cisco IOS Release 12.0(3)T and later supports the MPC+ features and protocols necessary to support IP. This chapter provides information about configuring CMPC+ support on the Channel Interface Processor (CIP) and Channel Port Adapter (CPA) types of Cisco Mainframe Channel Connection (CMCC) adapters on a Cisco router.

This information is described in the following sections:

For a complete description of the CMPC+ commands in this chapter, refer to the "CSNA, CMPC, and CMPC+ Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Overview

CMPC+ enables High Performance Data Transfer (HPDT). It allows TCP/IP connections to the host through CMCC adapters, using either the TCP/IP stack or the High Speed Access Services (HSAS) IP stack. CMPC+ offers the following support:

- Support for TCP/IP and HSAS Transmission Group (TG)
- Support for one IP stack per MPC+ group

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

- Support for one read subchannel and one write subchannel per CMPC+ group. The read subchannel and write subchannel in an MPC+ group can be on different physical channels.
- Support for up to 64 KB per I/O block.
- Supports the CIP and the CPA

Up to 64 MPC+ groups can be configured on a CMCC, depending on memory configuration. CMPC+ can coexist with CMPC, TCP/IP Offload, CLAW, TN3270, and CSNA features.

Figure 264 shows an MVS host with a TCP/IP stack and a Cisco router configured with CMPC+ and IP.

Figure 264 MVS Host with TCP/IP Stack and Cisco Router with CMPC+

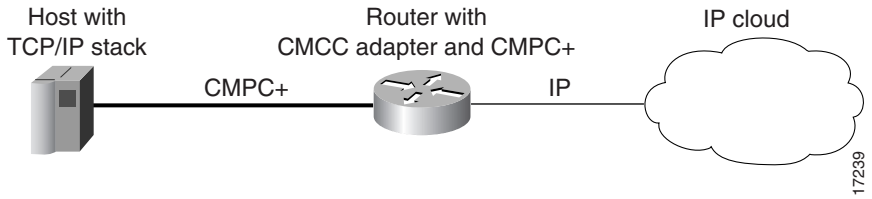
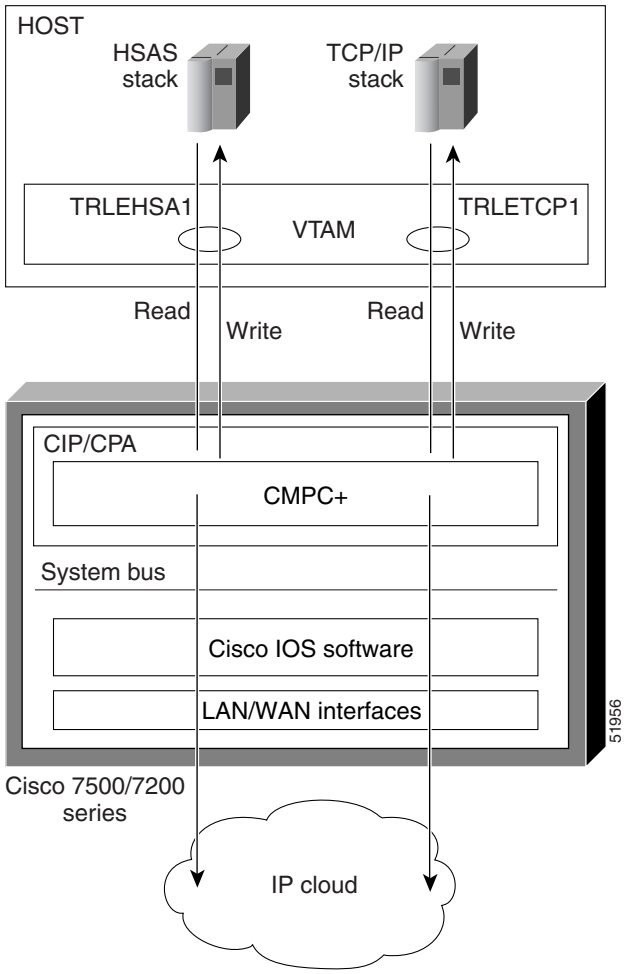


Figure 265 shows a scenario where a VTAM host is configured with both HSAS and TCP/IP stacks. Each stack on the host has a TG with a read and write subchannel. The CMCC adapter can be a CIP or CPA. On the outbound flow coming from the host, the Cisco router receives the MPC+ packets from the channel, removes the MPC+ headers, and transfers the packets as IP packets to the IP cloud. The router sends the packets to the appropriate interface, depending on the destination IP address in the packet.

Figure 265 CMPC+ IP Structure



On the inbound flow coming from the IP cloud via LAN or WAN interfaces, the IP packets are switched from the Cisco IOS software to the CMCC adapter and MPC+ headers are added to the packets. The MPC+ block of packets is then sent to the host.

IP communication from one IBM host to another can also be accomplished through the same CMCC adapter. Figure 266 illustrates IP packets going from Host 1 to Host 2 and from Host 2 to Host 1. These packets are routed through the same Route Switch Processor (RSP) in the Cisco router.

Figure 266 IP Communication Between IBM Hosts Through the Same CMCC

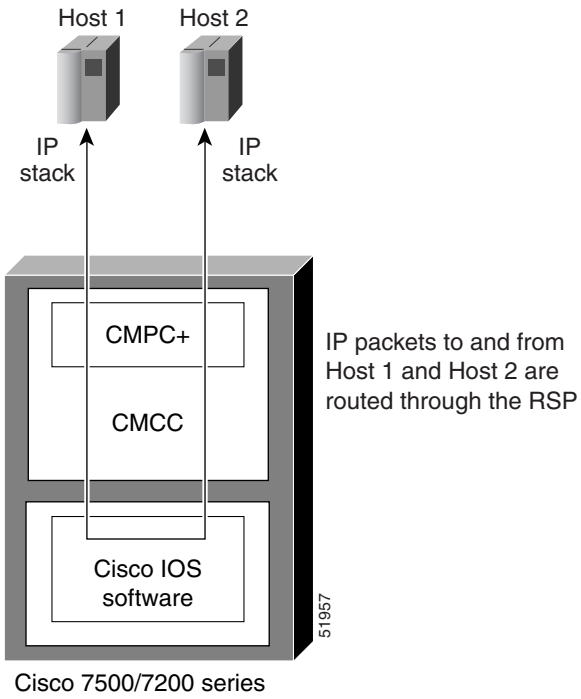


Figure 267 illustrates IP packets flowing between host 1 and host 2. IP packets can also be sent from one host to another host using different CMCC adapters.

Figure 267 IP Communication Between IBM Hosts Using Different CMCC Adapters

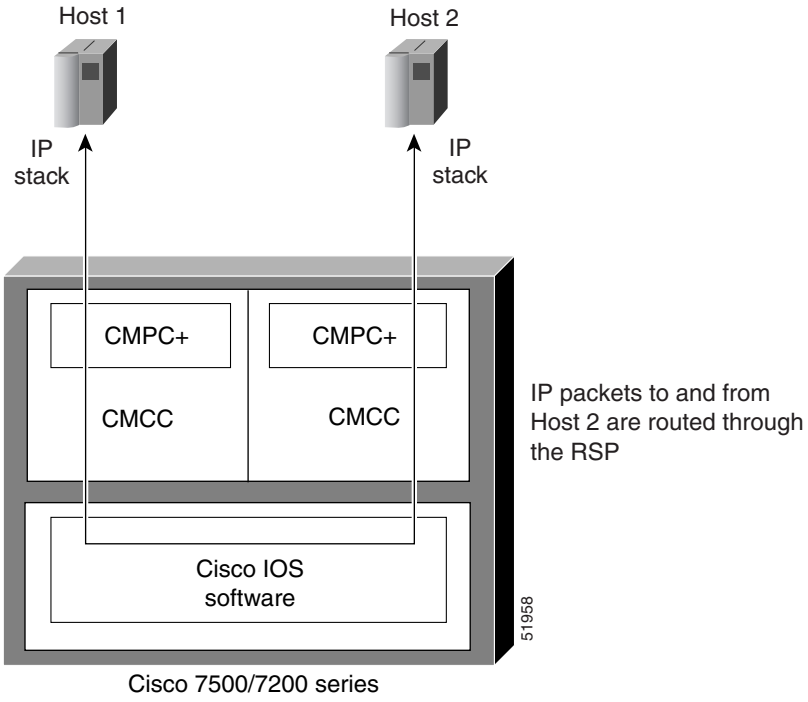


Figure 268 illustrates IP communication flowing between Host 1 and Host 2. IP communication can occur between two different hosts connected through two different routers.

Figure 268 IP Communication Between Hosts Through Different Routers

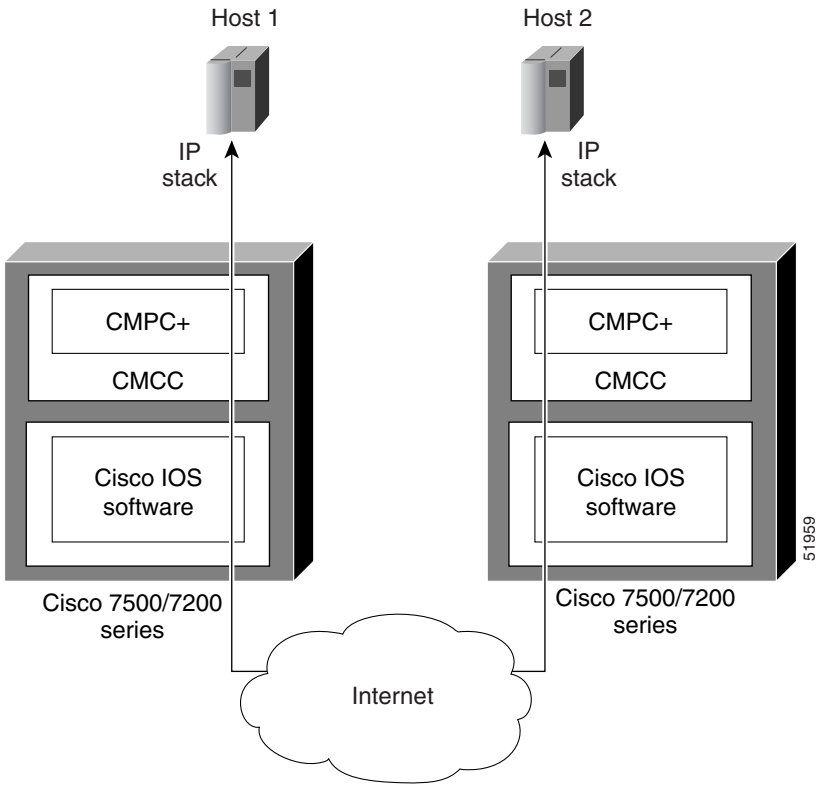
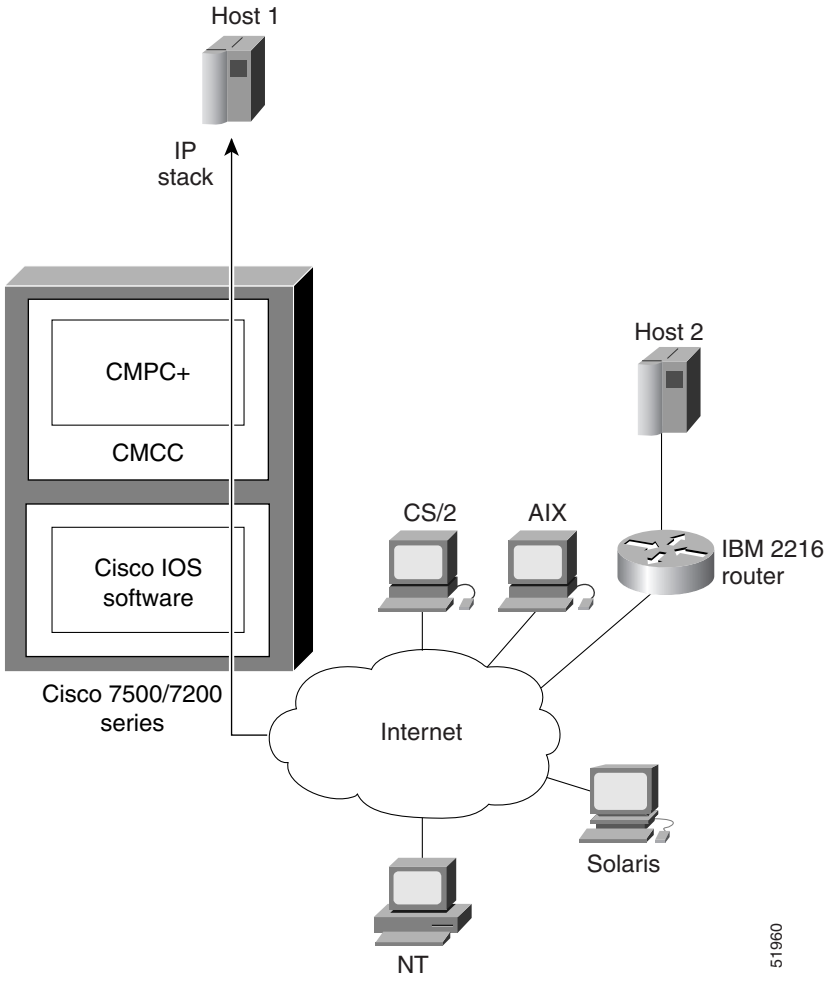


Figure 269 illustrates IP communication between a host and different IP nodes.

Figure 269 IP Communication from a Host to IP Nodes



Benefits

CMPC+ provides the following benefits:

- HPDT support
- IP connections to hosts using MPC+
- Coexistence with CMPC, CSNA, TN3270, TCP/IP Offload, and CLAW features on a CMCC
- ESCON and Parallel Channel support
- ESCON Director support
- EMIF support

Preparing to Configure CMPC+

The following topics in this section provide information that is useful when you are planning to configure CMPC+ support:

Hardware and Software Requirements

This section provides information about the router and mainframe requirements to support CMPC+.

Router Requirements

The CMPC+ feature is supported on the following router platforms and requires the appropriate CMCC microcode and a minimum of 32 MB DRAM on the CMCC adapter:

- Cisco 7500 series—Supports CIP adapters
- Cisco 7200 series—Supports ECPA and PCPA adapters
- Cisco 7000 series with RSP7000—Supports CIP adapters

You must configure the CMPC+ feature on the physical interface of a CMCC adapter. For a CIP, the physical interface is either 0 or 1. For the CPA adapters, ECPA and PCPA, the physical interface is port 0.

Mainframe Requirements

CMPC+ establishes channel connectivity to a S/390 mainframe host using the Virtual Telecommunications Access Method (VTAM) and IP stacks. The following software versions of S/390, VTAM, and IP stacks are required to configure CMPC+ on a CMCC adapter:

Operating System and VTAM Requirements

- OS/390 1.3 and later
- ACF/VTAM 4.4 and later

IP Stack Requirements

CMPC+ requires one of the following IP stacks on the mainframe:

- TCP/IP 3.3 and later
- HSAS

Mainframe Host Configuration Considerations

Configuring CMPC+ support requires that you perform tasks for configuration of the mainframe and the router sides of the network environment.

Often in the mixed network environment of mainframes and LANs, an MVS systems programmer installs and maintains the mainframe side of the network, while a network engineer manages the routers on the LAN side of the network. In such an environment, the successful configuration of CMPC+ support requires the close coordination between these job functions at a customer site.

This chapter contains information for both the network engineer and the MVS systems programmer to properly configure the network devices for CMPC+ support. The tasks for configuring CMPC+ support are organized by whether they are host-related configuration tasks or router-related configuration tasks. In addition, a topic for correlating the mainframe and router configuration is provided so that you can identify the dependencies between the host and router configuration elements and be sure that they are set up correctly.

Defining the Channel Subsystem for the Router

To establish the path and allocate the range of subchannel addresses that the CMCC adapter can use for the CMPC+ feature, you need to specify the channel subsystem definitions in the Input/Output Control Program (IOCP) or Hardware Configuration Definition (HCD).

For more information about the statements that might be defined in an IOCP file for parallel channels and ESCON channels on the CIP or CPA, see the "Defining the Channel Subsystem for the Router" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter of this publication.

Disabling the Missing Interrupt Handler

Because the appropriate configuration of the missing interrupt handler (MIH) varies according to the protocols and software releases used, Cisco offers the following guidance:

- For OS/390 releases Version 2 Release 4 and earlier, set the MIH to zero.
- For OS/390 releases later than Version 2 Release 4 and z/OS releases, refer to the following section of the z/OS Communications Server IP Configuration Reference: http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/f1a1b420/1.2.13?SHELF=f1a1bk31&DT=20020604120755#HDRMOLLY

For information about how to disable the MIH for the unit addresses being used for your CMCC adapter configuration, see the section "Disabling the Missing Interrupt Handler" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter of this publication.

Related Publications

The following mainframe-related publications might be useful when configuring the mainframe to support CMPC+ on a CMCC adapter in the router:

- *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00
- *OS/390 TCP/IP OpenEdition Planning and Release Guide*, SC31-8303-00
- *OS/390 TCP/IP OpenEdition User's Guide*, GC31-8305-00
- *IP Planning and Migration Guide*, SC31-8512
- *IP User's Guide*, C31-8514-00
- *High Speed Access Services User's Guide*, GC31-8676
- *ES/390 Principles of Operation*, SA22-7201
- *VTAM Network Implementation Guide*, SC31-8370-00
- *VTAM Resource Definition Reference*, SC31-8377-00
- *VTAM V4R4 Resource Definition Samples*, SC31-8378-00

Configuring CMPC+ Support

This section describes the configuration tasks that are required to install CMPC+ support on the mainframe and router and includes the following topics:

See the "CMPC+ Configuration Examples" section on page 33 for examples.

CMPC+ Configuration Guidelines

To configure the CMPC+ feature, you must configure the host VTAM parameters, the TCP/IP or HSAS stacks on the host, and the CMCC adapter. Consider the following guidelines as you prepare to configure CMPC+ support:

- A CMPC+ link uses two subchannels: one read and one write. Some IBM implementations of MPC+ allow multiple read and multiple write subchannels within a link. CMPC+ does not support multiple read and write subchannels. Only one read subchannel and one write subchannel can be configured for each CMPC+ link. A CMPC+ link is also referred to as a CMPC+ TG.
- On the router a CMPC+ TG consists of one read subchannel definition, one write subchannel definition, and a TG definition, associated by a unique *tg-name*.
- A CMCC adapter can have multiple CMPC+ links (TGs), up to a maximum of 64.
- To configure the TCP/IP or HSAS interface for the CMCC adapter, use the **tg** (CMPC+) command.
- To define the host subchannel (or path) and device, use the **cmpc** command on the router. One **cmpc** command defines the read subchannel, and one **cmpc** command defines the write subchannel. The **cmpc** command is configured on the CMCC adapter's physical interface (port 0 or 1 on a CIP; port 0 on a CPA).
- The two subchannels in a CMPC+ link do not need to be adjacent devices. Either channel can be the read subchannel or the write subchannel. The two subchannels can be on separate channel process IDs (CHPIDs) in the host.
- The two subchannels must be connected to the same CMCC adapter, however they do not have to be connected to the same physical channel interface on a CIP. On a CIP it is possible to connect a read subchannel to channel interface 0, while the write subchannel is connected to channel interface 1.
- The host IOCP or HCD parameters must coordinate with the **cmpc** command parameters on the router and the transport resource list major node definition to specify the subchannel path, device, and subchannel address.
- To configure MPC+ on the host, define the Transport Resource List (TRL). If you do not plan to support HPR, then you need to disable support in the TRL major node by configuring HPR=NO.
- CMPC+ can coexist with CLAW, TCP/IP offload, IP host backup, CSNA, CMPC, and TN3270 server features on the router.

CMPC+ Host Configuration Task List

This section contains the following host configuration tasks:

Configuring the VTAM Transport Resource List Major Node

The CMPC+ feature supports standard Transport Resource List (TRL) major node configurations, standard TCP Profiles and OE configuration statements. The host configuration must be coordinated with the CMPC+ configuration for IP connectivity to be successful.

To define the TRL, you must have two valid subchannel addresses configured in the IOCP or HCD on the host that can be used for the read and write subchannels. The read/write subchannels that you configure in the TRL should correlate with the unit addresses configured in the *device* argument of the **cmpc** commands. CMPC+ requires a unique TRLE for each CMPC+ TG.

The following sample configuration shows an example of a typical TRL major node configuration:

```
JECTRLG VBUILD TYPE=TRL
JCTRLG74 TRLE LNCTL=MPC,                                        X
               MAXBFRU=16,                                      X
               REPLYTO=25.5,                                    X
               MPCLEVEL=NOHPDT,                                 X
               READ=(274),                                      X
               WRITE=(275)
```

In this example, device 274 has been configured for read and 275 has been configured for write. Devices 274 and 275 must be available subchannels in the IOCP or HCD definition for the CMCC adapter connection.

You should activate the TRL before activating the corresponding local major node. The following example shows the command to activate a TRL, where the ID parameter specifies the name of the TRL:

```
v net,act,id=jectrlg,update=add
```

Note The argument “update=add” is preferred and is the default for later versions of VTAM. The argument “update=all” can cause inactive TRLEs to be deleted unexpectedly from ISTTRL. However, “update=all” must be used if you change an active TRL data set and want the changes to become active.

The following commands are useful for displaying the current list of TRLEs:

- **d net,trl**
- **d net,id=isttrl,e**
- **d net,trl,trle=trle_name**

For details on how to configure the TRL major node, see the following IBM documents:

- *VTAM Resource Definition Samples*, SC31-6554
- *VTAM Operation*, SC31-6549
- *VTAM Network Implementation Guide*, SC31-6548

Configuring the Stacks

This section provides samples for the following tasks:

- Configuring the IBM TCP/IP Stack, page 12
- Configuring the HSAS Stack, page 13

Configuring the IBM TCP/IP Stack

Following is an excerpt of a sample TCP/IP profile. The most important configuration commands are in bold:

```
DEVICE JCTRLG74 MPCPTP
LINK JECIP1   MPCPTP JCTRLG74
;
telnetparms
  TIMEMARK 600
  PORT 23
  DBCSTRANSFORM
endtelnetparms
;
ASSORTEDPARMS NOFWD ENDASSORTEDPARMS
;
PORT
    20 TCP OMVS      NOAUTOLOG ; FTP Server
    21 TCP OMVS                ; FTP Server
    23 TCP INTCLIEN            ; TELNET Server 3.4
    25 TCP SMTP                ; SMTP Server
    53 TCP NAMESRV             ; Domain Name Server
    53 UDP NAMESRV             ; Domain Name Server
   111 TCP OMVS                ; OE Portmap Server
   111 UDP OMVS                ; OE PORtmap Server
   135 UDP NCSLLBD             ; NCS Location Broker
   161 UDP SNMPD               ; SNMP Agent
   162 UDP SNMPQE              ; OE SNMPQE Agent
   515 TCP LPSERVE             ; LPD Server
   520 UDP OROUTED             ; OE RouteD Server
   750 TCP MVSKERB             ; Kerberos
   750 UDP MVSKERB             ; Kerberos
   751 TCP ADM@SRV             ; Kerberos Admin Server
   751 UDP ADM@SRV             ; Kerberos Admin Server
  2049 UDP MVSNFS              ; NFS Server
  3000 TCP CICSTCP             ; CICS Socket
HOME
    172.18.20.51   JECIP1
GATEWAY
    172.18.20.49 = JECIP1 32000 HOST
    DEFAULTNET  172.18.20.49  JECIP1 4468 0
;
; TRANSLATE
BEGINVTAM
 3278-2-E NSX32702 ; 24 line screen
 3279-2-E NSX32702 ; 24 line screen
 3278-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 lines
 3279-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 lines
 3278-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 lines
 3279-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 lines
 3278-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80
 3279-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80
DEFAULTLUS
      TCP20000..TCP20999  ; allow 1000 LU-LU SESSIONS on this TCPIP
  ENDDEFAULTLUS
```

```
ALLOWAPPL * ; Allow access to all applications
USSTCP USSTCPMG JECIP1
ENDVTAM
DATASETPREFIX TCPMVSG.TCPIP4
START JCTRLG74
```

For the CMCC adapter, the MTU size on the DEFAULTNET statement must be 4468 or less to ensure that the CMCC adapter does not receive packets larger than the CMCC adapter's MTU.

Configuring the HSAS Stack

Following are the sample OE configuration commands for configuring and activating the HSAS stack:

```
oeifconfig trle host-ip-addr router-ip-addr [mtu size] [netmask netmask]
oeifconfig trle [up|down|detach]
oeroute [flags] [add|delete] [default|dest-ip-addr gateway-ip-addr [metric]]
oenetopts [+-a [config-file-name]]
```

For HSAS, the MTU size must be set to 4468.

CMPC+ Router Configuration Task List

This section describes the configuration tasks associated with the CMPC+ feature.

- Assigning an IP Address to the Network Interface, page 13
- Configuring the CMPC+ Subchannels, page 14
- Configuring the CMPC+ TGs, page 15

Assigning an IP Address to the Network Interface

To assign an IP address to the network interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# ip address ip-address mask secondary`	Assigns an IP address to the network interface.

Configuring the CMPC+ Subchannels

Configuring the CMPC+ subchannels establishes the physical path between the CMCC interface and the mainframe channel.

To define a CMPC+ read subchannel and CMPC+ write subchannel, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure CMPC. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the physical interface, which is port 0 or 1. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# cmpc path device tg-name read`	Defines the CMPC+ read subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. • *tg-name*—Name of the CMPC+ TG, up to 8 characters.
Step 3	`Router(config-if)# cmpc path device tg-name write`	Defines the CMPC+ write subchannel device with the following arguments: • *path*—Four-digit value that represents the channel path for the device. The path value is always 0100 for parallel channels. • *device*—Unit address for the device on the subchannel. This unit address must be a different address than the unit address for the CMPC+ read subchannel. • *tg-name*—Name of the CMPC+ TG, up to eight characters.

Use the **no cmpc** *path device* command to remove the definition of a subchannel.

Mainframe Configuration Tips

- Configuring the subchannel information in the router requires that you correlate the *path* and *device* information from the IOCP or HCD file on the host.
  - The *path* argument is a four-digit hexadecimal value that concatenates the path value (two digits), EMIF partition number (one digit), and control unit logical address (one digit).
  - The *device* argument is a valid number in the UNITADD range of the IOCP CNTLUNIT statement for the CMPC internal LAN adapter.

  For detailed information about how to determine the *path* and *device* values for the **cmpc** command, see the "Correlating Channel Configuration Parameters" section in the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this guide.
- The **cmpc** commands on the router define the subchannel addresses that CMPC+ will use to connect to the host, and correspond to the definitions in the TRL major node on the host. Normally, the last two hexadecimal digits in the READ parameter of the TRL match the value of the *device* argument in the corresponding **cmpc read** command. Similarly, the last two hexadecimal digits in the WRITE parameter of the TRL match the value of the *device* argument in the **cmpc write** command.

Configuring the CMPC+ TGs

To define a CMPC+ TG by name, use the following command in interface configuration mode on a CIP virtual interface or a CPA physical interface:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the CMPC+ TG. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# tg tg-name [ip | hsas-ip] host-ip-addr local-ip-addr broadcast`	Defines the type of IP stack and the IP addresses for the CMPC+ connection with the following arguments: • *tg-name*—Name (up to 8 characters) of the TG. This name must match the name specified in the **cmpc** commands. • **ip**—TCP/IP stack connection for the TG. • **hsas-ip**—HSAS IP stack connection for the TG. • *host-ip-addr*—IP address of the channel-attached host using this TG. This address must match the IP address in the HOME statement of the host's TCP/IP profile, or the host address in the *source-IP-address* argument of the **oeifconfig** command for HSAS on the host. • *local-ip-addr*—IP address configured on the virtual interface of the CMCC adapter. This address must match the IP address for the router in the DEFAULTNET statement of the host's TCP/IP profile, or the router address in the *destination-IP-address* argument of the **oeifconfig** command for HSAS on the host. • **broadcast**—Routing updates enabled to the host.

Use the **no tg** command to remove a CMPC+ TG from the configuration, which will deactivate the named CMPC+ TG. To change any parameter of the **tg** statement, the statement must be removed by using the **no tg** *tg-name* command.

Router Configuration Tip

The *name* that you specify for the TG must match the name that you specify in the *tg-name* argument of the **cmpc** commands on the physical interface of the same CMCC adapter.

Mainframe Configuration Tips

- The IP address that you specify for the *host-ip-addr* argument must match one of the following host configuration parameters:
  - For TCP/IP profile configurations, the *host-ip-addr* address must match the IP address specified in the HOME statement at the host.
  - For HSAS stack configurations, the *host-ip-addr* address must match the IP address specified in the *source-IP-address* argument of the **oeifconfig** command at the host.
- The IP address that you specify for the *local-ip-addr* argument must match one of the following host configuration parameters:
  - For TCP/IP profile configurations, the *local-ip-addr* address must match the IP address specified in the DEFAULTNET statement at the host.
  - For HSAS stack configurations, the local-ip-addr address must match the IP address specified in the destination-IP-address argument of the oeifconfig command at the host.

Correlating the Mainframe and Router Configuration Elements

Table 17 shows a summary of the configuration elements on the router and host that must be correlated for proper operation of CMPC+. The column labeled "Configuration Element" identifies the type of entity to be configured. The columns labeled "Router Configuration" and "Mainframe Configuration" identify the related parameters on the router and the mainframe whose values must be compatible or match.

Table 17 Relationship of Configuration Elements for CMPC+

Configuration Element	Router Configuration	Mainframe Configuration
Subchannels	*path* and *device* arguments of the **cmpc** command	RESOURCE PARTITION, CHPID, and CNTLUNIT statements of the IOCP definition defining the following parameters for the CMPC+ channel path: • LPAR number (if defined) in the RESOURCE PARTITION and CHPID statements—Specify in the 3rd digit of the *path* argument in the router **cmpc** command. • CUADD value (if defined) in the CNTLUNIT statement—Specify in the 4th digit of the *path* argument in the router **cmpc** command. • Available device address in the UNITADD parameter of the CNTLUNIT statement—Specify in the *device* argument of the router **cmpc** command.
Read/write subchannels	*device* argument for the **cmpc read** command *device* argument for the **cmpc write** command	Subchannel for the READ parameter of the TRL major node Subchannel for the WRITE parameter of the TRL major node

Table 17 Relationship of Configuration Elements for CMPC+ (continued)

Configuration Element	Router Configuration	Mainframe Configuration
Host IP address	*host-ip-addr* argument for the **tg** (CMPC+) command	• For TCP/IP profile configuration—HOME statement • For HSAS configuration—*source-IP-address* argument of the **oeifconfig** command
Router IP address	*local-ip-addr* argument for the **tg** (CMPC+) command	• For TCP/IP profile configuration—DEFAULTNET statement • For HSAS configuration—*destination-IP-address* argument of the **oeifconfig** command

Verifying a CMPC+ Configuration Using TCP/IP

Configuring CMPC+ includes tasks for both the mainframe and the router. This section describes the steps to verify that you have successfully configured CMPC+ on a CIP. It provides procedures to verify connectivity from the router perspective and from the host perspective, and includes troubleshooting tips as a guide when the configuration verification fails.

This section includes the following topics:

Initial Host and Router Configuration Using the TCP/IP Stack

When using the TCP/IP stack, consider that you begin verification with the following configurations on the host and router:

TRL Major Node Definition

```
JECTRLG VBUILD TYPE=TRL
JCTRLG74 TRLE LNCTL=MPC,                                        X
               MAXBFRU=16,                                      X
               REPLYTO=25.5,                                    X
               MPCLEVEL=NOHPDT,                                 X
               READ=(274),                                      X
               WRITE=(275)                                      X
```

Note The verification procedures assume that the VTAM major node is defined, but not yet activated.

Host TCP/IP Stack Configuration in PROFILE.TCPIP

```
DEVICE JCTRLG74 MPCPTP
LINK JECIP1   MPCPTP JCTRLG74
;
telnetparms
  TIMEMARK 600
  PORT 23
  DBCSTRANSFORM
endtelnetparms
;
ASSORTEDPARMS NOFWD ENDASSORTEDPARMS
;
PORT
    20 TCP OMVS      NOAUTOLOG ; FTP Server
    21 TCP OMVS                ; FTP Server
    23 TCP INTCLIEN            ; TELNET Server 3.4
    25 TCP SMTP                ; SMTP Server
    53 TCP NAMESRV             ; Domain Name Server
    53 UDP NAMESRV             ; Domain Name Server
   111 TCP OMVS                ; OE Portmap Server
   111 UDP OMVS                ; OE PORtmap Server
   135 UDP NCSLLBD             ; NCS Location Broker
   161 UDP SNMPD               ; SNMP Agent
   162 UDP SNMPQE              ; OE SNMPQE Agent
   515 TCP LPSERVE             ; LPD Server
   520 UDP OROUTED             ; OE RouteD Server
   750 TCP MVSKERB             ; Kerberos
   750 UDP MVSKERB             ; Kerberos
   751 TCP ADM@SRV             ; Kerberos Admin Server
   751 UDP ADM@SRV             ; Kerberos Admin Server
  2049 UDP MVSNFS              ; NFS Server
  3000 TCP CICSTCP             ; CICS Socket
HOME
    172.18.20.51   JECIP1
GATEWAY
    172.18.20.49 = JECIP1 32000 HOST
    DEFAULTNET  172.18.20.49  JECIP1 4472 0
;
; TRANSLATE
BEGINVTAM
 3278-2-E NSX32702 ; 24 line screen
 3279-2-E NSX32702 ; 24 line screen
 3278-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 lines
 3279-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 lines
 3278-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 lines
 3279-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 lines
 3278-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80
 3279-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80
DEFAULTLUS
      TCP20000..TCP20999  ; allow 1000 LU-LU SESSIONS on this TCPIP
  ENDDEFAULTLUS
ALLOWAPPL * ; Allow access to all applications
USSTCP USSTCPMG JECIP1
ENDVTAM
DATASETPREFIX TCPMVSG.TCPIP4
START JCTRLG74
```

Initial Router Configuration on a CIP

```
interface channel 2/1
 no ip address
 no ip directed-broadcast
 no keepalive
```

```
!
interface channel 2/2
 ip address 172.18.20.49 255.255.255.248
 no ip redirects
 no ip directed-broadcast
 no keepalive
 tx-queue-limit 100
```

Note The initial router configuration shows the configuration prior to configuring the CMPC+ feature.

Verifying CMPC+ Channel Connectivity

If you have defined the channel paths for the router at the mainframe host in the IOCP or HCD, you can begin to configure the router for CMPC+ support and verify connectivity at the channel level first. Isolating this level of verification is useful when the VTAM configuration is not completed, but you want to establish that the router can successfully communicate with the host.

Verifying channel connectivity confirms the following aspects of the router configuration:

- Microcode is loaded on the CMCC
- CMCC adapter is functional
- CMCC can communicate with the host over the channel path

This section includes the following tasks:

- Verifying CMPC+ Channel Connectivity From the Router, page 19
- Verifying CMPC+ Channel Connectivity From the Host, page 21
- Troubleshooting Tips for Channel Connectivity, page 21

Verifying CMPC+ Channel Connectivity From the Router

The steps in this section show how to verify the CMPC+ channel configuration beginning with configuring the **cmpc** commands on the router's physical interface. The following assumptions are made for the procedure described in this section:

- The router's virtual interface is already configured with the statements as shown in the initial router configuration for a CIP shown in Figure 269.
- The router has the recommended CMCC hardware and microcode versions to support the CMPC feature. You can use the **show version**, **show controllers cbus**, and **show controllers channel** commands to verify the Cisco IOS software and CMCC microcode versions.

Note Before you begin on the router, run the **debug channel events** command so that you can verify the messages on the router console.

To verify CMPC+ channel connectivity, perform the following steps:

Step 1 From the router, configure the **cmpc** commands on the physical interface according to your site's requirements as shown in the following example:

```
interface channel 2/1
cmpc C190 74 MVSG-TCP READ
cmpc C190 75 MVSG-TCP WRITE
```

Confirm that you receive messages stating "Device Initialized," similar to the following displays:

```
PA1 MPC C190-74 Device initialized
PA1 MPC C190-75 Device initialized
```

Step 2 Configure the CMPC+ TG according to your site's requirements as shown in the following example:

```
interface channel 2/2
tg MVSG-TCP ip 172.18.20.51 172.18.20.49
```

Confirm that you receive a message stating that the CMPC+ TG is "Initialized," similar to the following display:

```
CMPC-TG MVSG-TCP initialized
```

Step 3 To verify that the channel is up and the line protocol is up, go to EXEC command mode and run the **show interfaces channel** command as shown in the following example:

```
show interfaces channel 2/1
```

Step 4 To verify that the physical channel is up, run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the path field in the output for the CMPC+ devices shows "ESTABLISHED," which means that the physical channel is up.

Step 5 If your **show** command output matches the values described in Step 3 and Step 4, then the channel connection between the mainframe and the router is established. If you cannot confirm the values, see the "Troubleshooting Tips for Channel Connectivity" section on page 21.

Verifying CMPC+ Channel Connectivity From the Host

After CMPC+ has been configured on the router, you can also verify channel connectivity from the host by performing the following steps:

Step 1 From the host, verify that the devices are online using the following sample command to display the device 274 for a range of two (or 274-275):

```
d u,,,274,2
```

Step 2 If the devices are offline, then vary the devices online according to your site's configuration as shown in the following sample commands:

```
v 274,online
v 275,online
```

Note The CHPID for the device should already be active on the host.

Step 3 If the devices come online, then the channel connection between the mainframe and the router is established. If the device does not come online, or you receive the message "No paths physically available," see the "Troubleshooting Tips for Channel Connectivity" section on page 21.

Troubleshooting Tips for Channel Connectivity

There are several indicators on the router and the mainframe that indicate that the channel connection is not available.

- From the router, you might see the following things:
  - The output from the **show interfaces channel** command shows that the channel or line protocol is down.
  - The output from the **show interface channel statistics** command shows that the path is not established (the physical channel is not up).
- From the host, you might see the following things:
  - The device is not online.
  - When you vary the device online, you receive the message "No paths physically available."

Recommended Actions

If you determine that the channel connection is not available, review the following tasks to be sure that you have performed them correctly:

- Be sure that you enabled the CMPC+ router configuration using the **no shut** command to restart the interface. If you configured both the physical and virtual interface on a CIP, be sure to run the **no shut** command on both interfaces.
- Be sure that the CMPC+ devices (and paths) are online at the host.
- Verify that the *path* and *device* arguments that you specified in your **cmpc** configuration command correlate properly to the host IOCP or HCD configuration.

If none of these recommended actions allow you to establish the channel connection, check your CMCC LED indicators and the physical channel connection.

Verifying Communication Between the Host and Router Using TCP/IP

After the VTAM TRL major node definition is installed, the TCP/IP stack is configured, and the router is configured, you can verify communication between the host and router.

This section includes the following verification procedures:

Verifying Communication From the Host Using TCP/IP

This procedure describes how to verify from the host that the VTAM TRL major node definition is configured and activated.

To verify communication with VTAM using CMPC+, perform the following steps:

Step 1 Activate the TRLE using the following sample command:

```
v net,act,id=JCTRLG74,update=add
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I ISTTRL ACTIVE
```

Step 2 Display the TRL status using the following command:

```
d net,trl
```

Verify that the TRLE is present but not active, as shown in the following console message:

```
IST1314I TRLE=JCTRLG74 STATUS=NEVAC CONTROL=MPC
```

Step 3 Start the TCP/IP task using the following sample command:

```
s TCPMVSG4
```

Verify that the TCP/IP task starts, the TRLE device and HSAS stack initializes and the interfaces are active as shown in the following console messages:

```
$HASP100 TCPMVSG4 ON STCINRDR
  IEF695I START TCPMVSG4 WITH JOBNAME TCPMVSG4 IS ASSIGNED TO USER
  OMVSKERN, GROUP OMVSGRP
  $HASP373 TCPMVSG4 STARTED
  IEF403I TCPMVSG4 - STARTED - TIME=11.33.21
  IEE252I MEMBER CTIEZB00 FOUND IN SYS1.OS390R7.PARMLIB
  EPW0250I EPWPITSK: FFST INITIALIZATION FOR TCP COMPLETE
  EZZ0300I OPENED PROFILE FILE DD:PROFILE
  EZZ0309I PROFILE PROCESSING BEGINNING FOR DD:PROFILE
  EZZ0316I PROFILE PROCESSING COMPLETE FOR FILE DD:PROFILE
  EZZ0334I IP FORWARDING IS DISABLED
  EZZ0335I ICMP WILL NOT IGNORE REDIRECTS
  EZZ0337I CLAWUSEDOUBLENOP IS CLEARED
  IEF196I IEF237I 0275 ALLOCATED TO TP0275
  EZZ0345I STOPONCLAWERROR IS DISABLED
  EZZ4202I OPENEDITION-TCP/IP CONNECTION ESTABLISHED FOR TCPMVSG4
  BPXF206I ROUTING INFORMATION FOR TRANSPORT DRIVER TCPMVSG4 HAS BEEN
```

```
INITIALIZED OR UPDATED.
IEF196I IEF237I 0274 ALLOCATED TO TP0274
EZZ4313I INITIALIZATION COMPLETE FOR DEVICE JCTRLG74
EZZ4324I CONNECTION TO 172.18.20.49 ACTIVE FOR DEVICE JCTRLG74
EZB6473I TCP/IP STACK FUNCTIONS INITIALIZATION COMPLETE.
EZAIN11I ALL TCPIP INTERFACES FOR PROC TCPMVSG4 ARE ACTIVE.
```

Verifying Communication From the Router Using TCP/IP

This procedure describes how to verify communication with the VTAM TRL major node and the TCP/IP stack from the router.

To verify communication with VTAM from the router, perform the following steps:

Step 1 Run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the following is displayed in these fields of the output for the CMPC+ devices:

- Path—The CMPC+ path is “ESTABLISHED,” which means that the physical channel is up.
- Con—The connection value is “Y,” which means that the subchannel is up and the CMPC+ connection is established between the router and the mainframe.

Step 2 To verify that the CMPC+ subchannels are active, run the **show extended channel cmpc** command as shown in the following example:

```
show extended channel 2/0 cmpc
```

Step 3 To verify the operational status and configuration of the CMPC+ TGs, run the **show extended channel tg** command as shown in the following example:

```
show extended channel 2/2 tg detailed MVSG-TCP
```

Step 4 When the TCP/IP task has been started at the host, verify that you receive a message stating that the connection is activated as shown in the following example:

```
MVSG-TCP: Tcp/Ip Connection Activated
```

Step 5 To confirm that you can establish a connection to the host, run the following **ping** command at the router console and specify the IP address that is configured in the HOME statement of the TCP/IP profile (Figure 269):

```
router# ping 172.18.20.51
```

Verify that you receive a successful response to the **ping** command as shown in the following example:

```
Sending 5, 100-byte ICMP Echos to 172.18.20.51, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/12/12 ms
```

For information about other commands that are useful when diagnosing or monitoring your CMPC+ connection, see the “Monitoring and Maintaining CMPC+” section on page 32.

Troubleshooting Tips from the Host Using TCP/IP

This section describes recommended actions for the following problems that might occur during verification of communication between the host and router.

From the host, you receive the following messages when you start the TCP/IP task:

```
$HASP100 TCPMVSG4 ON STCINRDR
  IEF695I START TCPMVSG4 WITH JOBNAME TCPMVSG4 IS ASSIGNED TO USER
  OMVSKERN, GROUP OMVSGRP
  $HASP373 TCPMVSG4 STARTED
  IEF403I TCPMVSG4 - STARTED - TIME=11.40.57
  IEE252I MEMBER CTIEZB00 FOUND IN SYS1.OS390R7.PARMLIB
  EPW0250I EPWPITSK: FFST INITIALIZATION FOR TCP COMPLETE
  EZZ0300I OPENED PROFILE FILE DD:PROFILE
  EZZ0309I PROFILE PROCESSING BEGINNING FOR DD:PROFILE
  EZZ0316I PROFILE PROCESSING COMPLETE FOR FILE DD:PROFILE
  EZZ0334I IP FORWARDING IS DISABLED
  EZZ0335I ICMP WILL NOT IGNORE REDIRECTS
  EZZ4308I ERROR: CODE=80103016 DURING ACTIVATION OF DEVICE JCTRLG74.
  DIAGNOSTIC CODE: 02
  EZZ0337I CLAWUSEDOUBLENOP IS CLEARED
  EZZ0345I STOPONCLAWERROR IS DISABLED
  EZZ4202I OPENEDITION-TCP/IP CONNECTION ESTABLISHED FOR TCPMVSG4
  BPXF206I ROUTING INFORMATION FOR TRANSPORT DRIVER TCPMVSG4 HAS BEEN
  INITIALIZED OR UPDATED.
  EZB6473I TCP/IP STACK FUNCTIONS INITIALIZATION COMPLETE.
  EZAIN11I ALL TCPIP INTERFACES FOR PROC TCPMVSG4 ARE ACTIVE.
  EZZ0400I TELNET/VTAM (SECOND PASS) BEGINNING FOR FILE: DD:PROFILE
  EZZ6025I TELNET SEARCH OF USSTCPMG FAILED, RC = 00000004 RSN = 00000004
  EZZ6003I TELNET LISTENING ON PORT    23
  EZZ0403I TELNET/VTAM (SECOND PASS) COMPLETE FOR FILE: DD:PROFILE
  EZZ6027I TELNET TRANSFORM INITIALIZATION FAILED, RC: FFFF
  EZZ6028I TELNET TRANSFORM HAS ENDED
```

Recommended Action

Verify that the TRL major node is active at the host.

Troubleshooting Tips from the Router Using TCP/IP

This section describes recommended actions for the following problems that might occur during verification of communication between the host and router.

Problem

From the router, the **ping** command to the host fails as shown in the following example:

```
Sending 5, 100-byte ICMP Echos to 172.18.20.51, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Recommended Actions

- Verify that the TRL major node is active at the host.
- Verify that the IP address for the router in the TCP/IP profile matches the local IP address that you specified in the router's **tg** configuration command.

Problem

From the router, you receive the following messages when the TCP/IP task is started at the host:

```
%CIP2-6-MSG: slot2 %MPC-6-CMPCP_CV_LOG: MVSG-TCP: Event/State
PRE_VcActInd/PRS_Reset
%CIP2-3-MSG: slot2 %MPC-3-CMPCP_CV_ERR2: Possible Config error:
  Rcvd Host Local IP: 172.18.20.51, Expected: 172.18.20.52
```

Note Ignore any other error messages that might follow these errors on the router console until you follow the recommended action to resolve the problem.

Recommended Action

Verify that the IP address for the host in the TCP/IP profile matches the remote IP address that you specified in the router's **tg** configuration command.

Problem

From the router, you receive the following messages when the TCP/IP task is started at the host:

```
%CIP2-6-MSG: slot2 %MPC-6-CMPCP_CV_LOG: MVSG-TCP: Event/State
PRE_VcActInd/PRS_Reset
%CIP2-3-MSG: slot2 %MPC-3-CMPCP_CV_ERR2: Possible Config error:
  Rcvd protocol: TCP/IP, Expected: HSAS/IP
```

Note Ignore any other error messages that might follow these errors on the router console until you follow the recommended action to resolve the problem.

Recommended Action

Verify that the router's **tg** configuration command specifies the **ip** protocol keyword and not **hsas-ip**.

Verifying a CMPC+ Configuration Using HSAS

Configuring CMPC+ includes tasks for both the mainframe and the router. This section describes the steps to verify that you have successfully configured CMPC+ on a CIP. It provides procedures to verify connectivity from the router perspective and from the host perspective, and includes troubleshooting tips as a guide when the configuration verification fails.

This section includes the following topics:

Initial Host and Router Configuration Using the HSAS Stack

When using the HSAS stack, consider that you begin verification with the following configurations on the host and router:

- TRL Major Node Definition, page 26
- Initial Router Configuration on a CIP, page 26

Note The verification procedures assume that the VTAM major node is defined, but not yet activated.

TRL Major Node Definition

```
HSASTRLG VBUILD TYPE=TRL
HSASTR72 TRLE  LNCTL=MPC,MAXBFRU=16,                                  X
               READ=(272),                                            X
               WRITE=(273)
```

Initial Router Configuration on a CIP

```
interface Channel2/1
 no ip address
 no ip directed-broadcast
 no keepalive
!
interface Channel2/2
 ip address 172.18.20.49 255.255.255.248
 no ip redirects
 no ip directed-broadcast
 no keepalive
 tx-queue-limit 100
```

Note The initial router configuration shows the configuration prior to configuring the CMPC+ feature.

Verifying CMPC+ Channel Connectivity

If you have defined the channel paths for the router at the mainframe host in the IOCP or HCD, you can begin to configure the router for CMPC+ support and verify connectivity at the channel level first. Isolating this level of verification is useful when the VTAM configuration is not completed, but you want to establish that the router can successfully communicate with the host.

Verifying channel connectivity confirms the following aspects of the router configuration:

- Microcode is loaded on the CMCC
- CMCC adapter is functional
- CMCC can communicate with the host over the channel path

Verifying CMPC+ Channel Connectivity From the Router

The steps in this section show how to verify the CMPC+ channel configuration beginning with running the **cmpc** command on the router's physical interface. The following assumptions are made for the procedure described in this section:

- The router's virtual interface is already configured with the statements as shown in the initial router configuration for a CIP shown in Figure 269.
- The router has the recommended CMCC hardware and microcode versions to support the CMPC feature. You can use the **show version**, **show controllers cbus**, and **show controllers channel** commands to verify the Cisco IOS software and CMCC microcode versions.

Note Before you begin on the router, run the **debug channel events** command so that you can verify the messages on the router console.

To verify CMPC+ channel connectivity, perform the following steps:

Step 1 From the router, configure the **cmpc** commands on the physical interface according to your site's requirements as shown in the following example:

```
interface channel 2/1
cmpc C190 72 MVSG-HSA READ
cmpc C190 73 MVSG-HSA WRITE
```

Confirm that you receive messages stating "Device Initialized," similar to the following displays:

```
PA1 MPC C190-72 Device initialized
PA1 MPC C190-73 Device initialized
```

Step 2 Configure the CMPC+ TG according to your site's requirements as shown in the following example:

```
interface channel 2/2
tg MVSG-HSA ip 172.18.20.50 172.18.20.49
```

Confirm that you receive a message stating that the CMPC+ TG is "Initialized," similar to the following display:

```
CMPC-TG MVSG-HSA initialized
```

Step 3 To verify that the channel is up and the line protocol is up, go to EXEC command mode and run the **show interfaces channel** command as shown in the following example:

```
show interfaces channel 2/1
```

Step 4 To verify that the physical channel is up, run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the path field in the output for the CMPC+ devices shows "ESTABLISHED," which means that the physical channel is up.

Step 5 If your **show** command output matches the values described in Step 3 and Step 4, then the channel connection between the mainframe and the router is established. If you cannot confirm the values, see the "Troubleshooting Tips for Channel Connectivity" section on page 28.

Verifying CMPC+ Channel Connectivity From the Host

After CMPC+ has been configured on the router, you can also verify channel connectivity from the host by performing the following steps:

Step 1 From the host, verify that the devices are online using the following sample command to display the device 272 for a range of two (or 272-273):

```
d u,,,272,2
```

Step 2 If the devices are offline, then vary the devices online according to your site's configuration as shown in the following sample commands:

```
v 272,online
v 273,online
```

Note The CHPID for the device should already be active on the host.

Step 3 If the devices come online, then the channel connection between the mainframe and the router is established. If the device does not come online, or you receive the message "No paths physically available," see the "Troubleshooting Tips for Channel Connectivity" section on page 28.

Troubleshooting Tips for Channel Connectivity

There are several indicators on the router and the mainframe that indicate that the channel connection is not available.

- From the router, you might see the following things:
  - The output from the **show interfaces channel** command shows that the channel or line protocol is down.
  - The output from the **show interface channel statistics** command shows that the path is not established (the physical channel is not up).
- From the host, you might see the following things:
  - The device is not online.
  - When you vary the device online, you receive the message "No paths physically available."

Recommended Actions

If you determine that the channel connection is not available, review the following tasks to be sure that you have performed them correctly:

- Be sure that you enabled the CMPC+ router configuration using the **no shut** command to restart the interface. If you configured both the physical and virtual interface on a CIP, be sure to run the **no shut** command on both interfaces.
- Be sure that the CMPC+ devices (and paths) are online at the host.
- Verify that the *path* and *device* arguments that you specified in your **cmpc** configuration command correlate properly to the host IOCP or HCD configuration.

If none of these recommended actions allow you to establish the channel connection, check your CMCC LED indicators and the physical channel connection.

Verifying Communication Between the Host and Router Using HSAS

After the VTAM TRL major node definition is installed and the router is configured, you can configure the HSAS stack on the host and verify communication between the host and router.

This section includes the following verification procedures:

- Verifying Communication From the Host Using HSAS, page 29
- Verifying Communication From the Router Using HSAS, page 30
- Troubleshooting Tips From the Router Using HSAS, page 31

Verifying Communication From the Host Using HSAS

This procedure describes how to verify from the host that the VTAM TRL major node definition is configured and activated.

To verify communication with VTAM using CMPC+, perform the following steps:

Step 1 Activate the TRLE using the following sample command:

```
v net,act,id=HSASTR72,update=add
```

Verify that you receive the following console messages:

```
IST097I VARY ACCEPTED
IST093I ISTTRL ACTIVE
```

Step 2 Display the TRL status using the following command:

```
d net,trl
```

Verify that the TRLE is present but not active, as shown in the following console message:

```
IST1314I TRLE=HSASTR72 STATUS=NEVAC CONTROL=MPC
```

Step 3 From the TSO/ISPF menu at the host, enter Open Edition by typing **L8.4**.

Step 4 To configure the HSAS stack on the host, type the **oeifconfig** command as shown in the following example:

```
oeifconfig hsastr72 172.18.20.50 172.18.20.49 mtu 4468
```

Step 5 Display the HSAS configuration using the following command:

```
oenetstat -r
```

Verify that you receive a display similar to the following messages:

```
Destination     Gateway         Flags  Refs        Use Unreach Interface
172.18.20.49    172.18.20.49    UH        0          0       0 HSASTR72
172.18.20.50    *               UHL       0          0       0 HSASTR72
127.0.0.0       *               UL        0          0       0 LO0
```

Verifying Communication From the Router Using HSAS

This procedure describes how to verify communication with the VTAM TRL major node and the HSAS stack from the router.

To verify communication with VTAM from the router, perform the following steps:

Step 1 Run the **show extended channel statistics** command as shown in the following example:

```
show extended channel 2/1 statistics
```

Verify that the following is displayed in these fields of the output for the CMPC+ devices:

- Path—The CMPC+ path is "ESTABLISHED," which means that the physical channel is up.
- Con—The connection value is "Y," which means that the subchannel is up and the CMPC+ connection is established between the router and the mainframe.

Step 2 To verify that the CMPC+ subchannels are active, run the **show extended channel cmpc** command as shown in the following example:

```
show extended channel 2/0 cmpc
```

Step 3 To verify the operational status and configuration of the CMPC+ TGs, run the **show extended channel tg** command as shown in the following example:

```
show extended channel 2/2 tg detailed MVSG-HSA
```

Step 4 When the HSAS task is configured at the host, verify that you receive a message stating that the connection is activated as shown in the following example:

```
MVSG-HSA: Hsas/Ip Connection Activated
```

Step 5 To confirm that you can establish a connection to the host, run the following **ping** command at the router console and specify the IP address that is configured in the oeifconfig command at the host:

```
router# ping 172.18.20.50
```

Verify that you receive a successful response to the **ping** command as shown in the following example:

```
Sending 5, 100-byte ICMP Echos to 172.18.20.50, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/12/16 ms
```

For information about other commands that are useful when diagnosing or monitoring your CMPC+ connection, see the "Monitoring and Maintaining CMPC+" section on page 32.

Troubleshooting Tips From the Router Using HSAS

This section describes recommended actions for the following problems that might occur during verification of communication between the host and router.

Problem

From the router, the `Hsas/Ip Connection Activated` message does not appear and the **ping** command times out.

Recommended Action

- Verify that the IP address for the host in the **oeifconfig** command at the host matches the remote IP address that you specified in the router's **tg** configuration command.
- Verify that the TRL activated successfully at the host.

Problem

From the router, you receive the following messages when the **oeifconfig** command is run at the host:

```
%CIP2-6-MSG: slot2 %MPC-6-CMPCP_CV_LOG: MVSG-HSA: Event/State
PRE_VcActInd/PRS_Reset
%CIP2-3-MSG: slot2 %MPC-3-CMPCP_CV_ERR2: Possible Config error:
  Rcvd Local Addr: 172.18.20.49, Expected: 172.18.20.51
```

Note Ignore any other error messages that might follow these errors on the router console until you follow the recommended action to resolve the problem.

Recommended Action

Verify that the IP address for the router in the **oeifconfig** command matches the local IP address that you specified in the router's **tg** configuration command.

Problem

From the router, you receive the following messages when the **oeifconfig** command is run at the host:

```
%CIP2-6-MSG: slot2 %MPC-6-CMPCP_CV_LOG: MVSG-HSA: Event/State
PRE_VcActInd/PRS_Reset
%CIP2-3-MSG: slot2 %MPC-3-CMPCP_CV_ERR2: Possible Config error:
  Rcvd protocol: HSAS/IP, Expected: TCP/IP
```

Note Ignore any other error messages that might follow these errors on the router console until you follow the recommended action to resolve the problem.

Recommended Action

Verify that the router's **tg** configuration command specifies the **hsas-ip** protocol keyword and not **ip**.

Monitoring and Maintaining CMPC+

The following topics in this section provide information about the different commands that you can use to monitor and maintain the CMCC interfaces that are configured for CMPC+:

Monitoring Interface Status

To monitor CMCC adapter interface status, you can display information about the interface, including the version of the software and the hardware, the controller status, and statistics about the interfaces. In addition, you can display information about feature-related statistics on the CMCC adapter. This section lists some additional commands that are useful when monitoring CMCC adapter interfaces that are configured for CMPC+.

For a complete list of the **show** commands that are related to monitoring CMCC adapter interfaces, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this guide. To display the full list of **show** commands, enter **show ?** at the EXEC prompt.

To display information related to CMPC+ configurations, use the following commands in EXEC mode:

Command	Purpose
`Router# show extended channel slot/port cmgr [tg-name]`	Displays information about the MPC+ TG connection manager.
`Router# show extended channel slot/port cmpc [path [device]]`	Displays information about each CMPC+ (and CMPC) subchannel configured on the specified CMCC adapter interface.
`Router# show extended channel slot/port tg [oper | stats] [detailed] [tg-name]`	Displays configuration, operational, and statistics information for CMPC+ (and CMPC) TGs configured on a specified CMCC adapter internal LAN interface.

Clearing Counters for CMPC+

You can reset the statistics counters that are displayed in the output of the **show extended channel** commands. You can reset the counters associated with an interface or a particular feature on the interface. If you are monitoring a particular threshold or statistic for a CMPC+ TG and need to reset a related counter, you can clear all those counters related to the TG.

For information about clearing other counters on the CMCC adapter interface, see the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this guide.

To clear the TG counters associated with CMPC+ on the CMCC adapters, use the following command in privileged EXEC mode:

Command	Purpose
`Router# clear extended counters channel slot/port tg`	Clears counters for statistics associated with TGs in the CMPC+ (and CMPC) features on the specified *slot/port*. The port value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to the physical interface, which is port 0.

Note This command will not clear counters retrieved using Simple Network Management Protocol (SNMP), but only those seen with the EXEC **show extended channel tg** command.

CMPC+ Configuration Examples

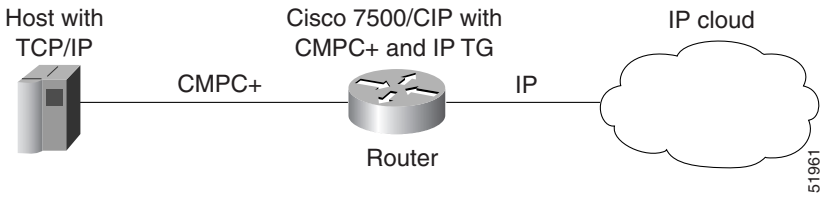
Following are the CMPC+ configuration examples shown in this section:

- CMPC+ with TCP/IP Stack Example, page 33
- CMPC+ with HSAS Stack Example, page 35
- CMPC+ with TCP/IP and HSAS Stacks Example, page 36
- CMPC+ TG Using Two Interfaces Example, page 39

CMPC+ with TCP/IP Stack Example

Figure 270 illustrates TCP/IP link for CMPC+ between a host and a Cisco router with a CMCC adapter. The configurations for this example follow.

Figure 270 CMPC+ IP with TCP/IP



Router

The following configuration is for the CIP in the Cisco 7500 router:

```
hostname ipclust1
!
microcode CIP flash slot0:cip27-4
microcode reload
!
```

```
interface Channel0/1
 no ip address
 no keepalive
 cmpc 0170 00 TG00 READ
 cmpc 0170 01 TG00 WRITE
!
interface Channel0/2
 ip address 80.12.165.1 255.255.255.0
 no ip redirects
 no ip directed-broadcast
 ip route-cache same-interface
 no ip mroute-cache
 load-interval 30
 no keepalive

 tg TG00     ip 80.12.165.2 80.12.165.1
```

In this configuration, the CMPC+ configuration is for the TCP/IP stack on the host. The host IP address of 80.12.165.2 in the TG statement corresponds to the IP address for the TCP/IP stack in the TCP/IP profile on the host. The IP address for the CIP is 80.12.165.2.

TCP/IP Profile

The following sample shows the TCP/IP Profile on the host:

```
ARPAGE 5
telnetparms timemark 600 port 23 dbcstransform endtelnetparms
ASSORTEDPARMS NOFWD ENDASSORTEDPARMS
;
DEVICE mpc4b00  MPCPTP
LINK MPCPLNK2 MPCPTP mpc4b00
;
AUTOLOG
  OEFTPE3
ENDAUTOLOG
INCLUDE TODD.MPCP.TCPIP.PROFILES(PORTS)
HOME
  80.12.165.2   MPCPLNK2
GATEWAY
; NETWORK    FIRST     DRIVER    PACKET   SUBNet mask    subnet value
;             HOP                 SIZE
 80.12.165.1   =    mpcplnk2     4468     host
DEFAULTNET 80.12.165.1 mpcplnk2  4468     0
BEGINVTAM


    ; Define logon mode tables to be the defaults shipped with the latest
    ; level of VTAM
  3278-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 line screen
  3279-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 line screen
  3278-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 line screen
  3279-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 line screen
  3278-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80 columns
  3279-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80 columns
    ; Define the LUs to be used for general users
  DEFAULTAPPL ECHOMVSE
; DEFAULTAPPL ECHOMVSE 10.10.1.188
; DEFAULTAPPL NETTMVSE
  DEFAULTLUS
      TCPE0000..TCPE9999
  ENDDEFAULTLUS
  ALLOWAPPL * ; Allow all applications that have not been previously
              ; specified to be accessed
ENDVTAM
```

```
DATASETPREFIX TODD.MPCP
start mpc4b00
```

In this TCP/IP profile, the DEVICE specifies the VTAM TRLE mpc4b00 and LINK specifies the link name (MPCPLNK2) associated with the IP address (80.12.165.2) for that link. The host IP address 80.12.165.2 that is specified for the TG in the router configuration must be identical to the IP address specified for the TG in the router configuration.

TRL Major Node

The following sample shows the TRL major node example:

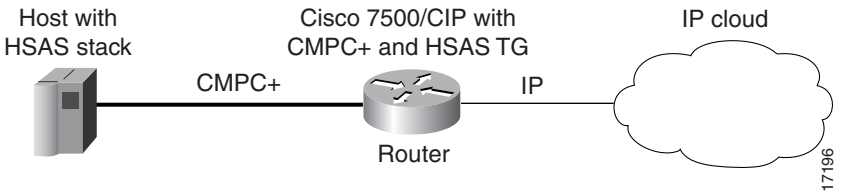
```
TRL4B00 VBUILD TYPE=TRL
MPC4B00  TRLE  LNCTL=MPC,MAXBFRU=16,                                    X
               READ=(4B00),                                            X
               WRITE=(4B01)
```

In this TRL major node example, the parameter *MPC4B00* must be identical to the LINK parameter in the TCP/IP profile.

CMPC+ with HSAS Stack Example

Figure 271 illustrates HSAS for CMPC+ between a host and a Cisco router with a CMCC adapter. The configurations for this example follow.

Figure 271 CMPC+ IP with HSAS



Router

The following sample shows the configuration for the CIP in the Cisco 7500 router:

```
hostname ipclust1
!
microcode CIP flash slot0:cip27-4
microcode reload
!
interface Channel0/1
 no ip address
 no keepalive
 cmpc 0170 02 TG00 READ
 cmpc 0170 03 TG00 WRITE
!

interface Channel0/2
 ip address 80.12.165.1 255.255.255.0
 no ip redirects
 no ip directed-broadcast
 ip route-cache same-interface
 no ip mroute-cache
 load-interval 30
 no keepalive
 tg TG00 hsas-ip 80.12.165.2 80.12.165.1
```

In this configuration, the CMPC + configuration is for the HSAS stack on the host. The IP address of 80.12.165.2 on the CIP corresponds to the HSAS IP address for the HSAS stack on the host.

Stack Definition in the OE Parmlib Member for HSAS

The following example shows the HSAS communications stack defined through the BPXPRMxy member in SYS1.PARMLIB. The most important configuration statement is shown in bold.

```
SYS1.PARMLIB(BPXPRMxy)
.
.
.
FILESYSTYPE    ENTRYPOINT(BPXTCINT)  TYPE(CINET)
NETWORK DOMAINNAME(AF_INET) DOMAINNUMBER(2) MAXSOCKETS(11000)
    TYPE(CINET)  INADDRANYPORT(5000)  INADDRANYCOUNT(4000)
SUBFILESYSTYPE NAME(TCPMVSE1) ENTRYPOINT(EZBPFINI) TYPE(CINET)
     DEFAULT
SUBFILESYSTYPE NAME(TCPMVSE2) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE3) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE4) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE5) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE6) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE7) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(TCPMVSE8) ENTRYPOINT(EZBPFINI) TYPE(CINET)
SUBFILESYSTYPE NAME(OESTACK)  ENTRYPOINT(BPXUIINT) TYPE(CINET)
```

The OpenEdition Common INET physical file system must be defined to include the HSAS communications stack (OESTACK). Whereas other IP communications stacks require a separate address space, the HSAS communications stack resides within the OpenEdition kernel.

The SUBFILESTYPE NAME(OESTACK) statement defines the HSAS stack. This statement must be coded exactly as shown in the example and only one such statement must be coded.

HSAS Configuration

The following sample is the HSAS configuration on the MVS host:

```
oeifconfig mpc4b02 80.80.165.2 80.12.165.1 mtu 4468
oeroute add default 80.12.165.1
```

TRL Major Node

The following sample is the TRL major node configuration:

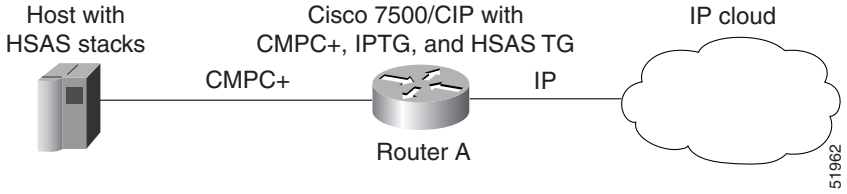
```
TRL4B02 VBUILD TYPE=TRL
MPC4B02 TRLE  LNCTL=MPC,MAXBFRU=16,                                  X
               READ=(4B02),                                          X
               WRITE=(4B03)
```

In this TRL major node configuration, the parameter *MPC4B02* must be identical to the TRLE parameter in the *oeifconfig* configuration.

CMPC+ with TCP/IP and HSAS Stacks Example

Figure 272 illustrates CMPC+ used between a Cisco router with a CMCC adapter and a host with both TCP/IP and HSAS stacks.

Figure 272 CMPC+ with TCP/IP and HSAS Stacks



Router

The following is the CMPC+ configuration for TCP/IP and HSAS in the CPA:

```
hostname ipclust2
!
enable secret 5 $1$2Py5$GmguUvRGwLdOj2UCi6cv71
enable password lab
!
microcode ecpa slot0:xcpa216-0.plus110698
microcode reload
ip subnet-zero
ip host bizarre 171.69.160.37
ip domain-name cisco.com
ip name-server 171.69.161.21
!


interface Channel5/0
 ip address 80.80.2.1 255.255.255.0
 no ip directed-broadcast
 ip route-cache same-interface
 no ip mroute-cache
 load-interval 30
 no keepalive
 cmpc 0180 00 TG00 READ
 cmpc 0180 01 TG00 WRITE
 cmpc 0180 02 TG02 READ
 cmpc 0180 03 TG02 WRITE
 tg TG00     ip 80.80.2.2 80.80.2.1
 tg TG02     hsas-ip 80.80.2.3 80.80.2.1
```

In this configuration, the cmpc configuration is for the TCP/IP and HSAS stacks on the host. The IP address of 80.80.2.2 corresponds to the IP address for the TCP/IP stack in the TCP/IP profile on the host. The IP address of 80.80.2.3 corresponds to the IP address in the HSAS stack on the host.

TRL Major Node for TCP/IP Stack

The following sample shows the TRL major node configuration for the TCP/IP stack:

```
TRL4900 VBUILD TYPE=TRL
MPC4900 TRLE  LNCTL=MPC,MAXBFRU=6,REPLYTO=3,                                  x
               READ=(4900),WRITE=(4901)
```

In this TRL major node example, the parameter *MPC4900* must be identical to the TRLE parameter in the LINK statement in the TCP/IP profile.

TRL Major Node for HSAS Stack

The following sample shows the TRL major node configuration for the HSAS stack:

```
TRL4902  VBUILD TYPE=TRL
MPC4902  TRLE  LNCTL=MPC,MAXBFRU=9,REPLYTO=3,                                 X
               READ=(4902),                                                   X
```

```
                WRITE=(4903)
```

In this TRL major node example, the parameter *MPC4B02* must be identical to the *oeifconfig* statement in the TCP/IP profile.

TCP/IP Profile

The following example shows the TCP/IP profile on the host:

```
ARPAGE 5
telnetparms timemark 600 port 23 dbcstransform endtelnetparms
ASSORTEDPARMS NOFWD ENDASSORTEDPARMS
;
DEVICE mpc4900  MPCPTP
LINK MPCPLNK2 MPCPTP mpc4900
;
INCLUDE TODD.MPCP.TCPIP.PROFILES(PORTS)
HOME
  80.80.2.2     MPCPLNK2
GATEWAY
; NETWORK    FIRST     DRIVER    PACKET   SUBNet mask    subnet value
;             HOP                 SIZE
 80.80.2.1     =       mpcplnk2   4468     host
DEFAULTNET 80.80.2.1   mpcplnk2   4468      0


BEGINVTAM
    ; Define logon mode tables to be the defaults shipped with the latest
    ; level of VTAM
 3278-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 line screen
 3279-3-E NSX32703 ; 32 line screen - default of NSX32702 is 24 line screen
 3278-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 line screen
 3279-4-E NSX32704 ; 48 line screen - default of NSX32702 is 24 line screen
 3278-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80 columns
 3279-5-E NSX32705 ; 132 column screen - default of NSX32702 is 80 columns
    ; Define the LUs to be used for general users
 DEFAULTAPPL ECHOMVSF
; DEFAULTAPPL NETTMVSE
 DEFAULTLUS
     TCPF0000..TCPF9999
  ENDDEFAULTLUS
  ALLOWAPPL * ; Allow access to all applications not previously specified
ENDVTAM
DATASETPREFIX TODD.MPCP
START mpc4900
```

In this TCP/IP profile, the DEVICE specifies the VTAM TRLE mpc4900 and LINK specifies the link name (MPCPLNK2) associated with the IP address (80.8.2.2) for that link. The IP address 80.80.2.1 must be identical to the IP address specified for the TG in the router configuration.

HSAS Stack Configuration

The following example shows the OE commands for the HSAS stack configuration:

```
oeifconfig mpc4902 80.80.2.3 80.80.2.1 mtu 4468
oeroute add default 80.80.2.1
```

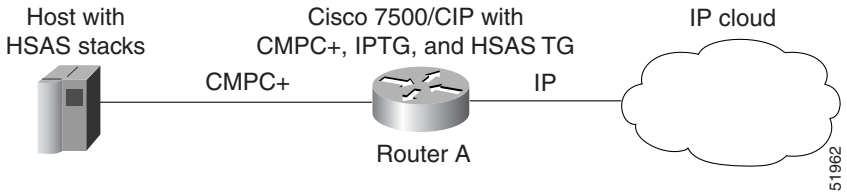
In this configuration, *mpc4902* must be identical to the TRLE parameter in the *oeifconfig* configuration.

CMPC+ TG Using Two Interfaces Example

Figure 273 illustrates CMPC+ used between a Cisco router with a CIP and a host with split TGs.

Note The split TG configuration is only supported on a CIP with two physical interfaces.

Figure 273 CMPC+ with Split TG



Router

The following example shows the CMPC+ router configuration for a split TG:

```
interface Channel0/0
 no ip address
 no ip directed-broadcast
 no keepalive
 cmpc 0170 00 TG00 READ
!
interface Channel0/1
 no ip address
 no ip directed-broadcast
 no keepalive
 cmpc 0170 00 TG00 WRITE
!
interface Channel0/2
 ip address 80.12.165.1 255.255.255.0
 no ip redirects
 no ip directed-broadcast
 ip route-cache same-interface
 no ip mroute-cache
 load-interval 30
 no keepalive
 tg TG00      hsas-ip 80.12.165.2 80.12.165.1
```

HSAS Configuration

The following example shows the HSAS configuration on the host:

```
oeifconfig mpcsplit 80.12.165.2 80.12.165.1 mtu 4468
oeroute add default 80.12.165.1
```

TRL Major Node for HSAS Stack

The following example shows the TRL major node configuration for the HSAS stack:

```
TRLSPLIT VBUILD TYPE=TRL
MPCSPLIT TRLE  LNCTL=MPC,MAXBFRU=16,REPLYTO=3,                          X
               READ=(5200),                                             X
               WRITE=(4B00)
```

In this TRL major node example, the parameter *mpcsplit* must be identical to the TRLE parameter in the LINK statement within the *oeifconfig* statement.

Configuring the TN3270 Server

The implementation of TN3270 Server on a channel-attached router using the CIP or CPA provides an effective method of removing the processing of TN3270 sessions from valuable mainframe cycles to a faster and more efficient router. This chapter provides information about configuring TN3270 Server support on the CIP and CPA types of CMCC adapters on a Cisco router.

This information is described in the following sections:

For general information about configuring CMCC adapters, refer to the "Configuring Cisco Mainframe Channel Connection Adapters" chapter in this publication.

For a complete description of the TN3270 server commands in this chapter, refer to the "TN3270 Server Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2). To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Platform Support for Cisco IOS Software Features" section on page li in the "Using Cisco IOS Software" chapter.

Overview

This section provides a brief introduction to the environments where the TN3270 server feature is used and describes some of the primary benefits and functions of the TN3270 server.

The following sections in this topic provide background information about the TN3270 Server:

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

Additional details about the TN3270 Server implementation can be found in the *TN3270 Design and Implementation Guide* available on Cisco.com.

Benefits

The latest release of the TN3270 Server feature on the CMCC implements RFC 2355, *TN3270 Enhancements* and RFC 2562, *Definitions of Protocol and Managed Objects for TN3270E Response Time Collection Using SMIv2 (TN3270E-RT-MIB).*

The TN3270 server provides the following benefits:

- Supports clients using the ASSOCIATE request.
- Maintains knowledge of printer and terminal relationships when an association is defined between LU resources.
- Enables clients to acquire a terminal LU and its associated printer without desktop configuration to specific LUs by grouping LUs in clusters.
- Enables you to capture response-time statistics for individual sessions and clients or for groups of sessions and clients.
- Supports specification of LU names for dynamic definition of dependent LUs (DDDLUs).
- Controls how keepalives are generated and keepalive responses are handled by the CMCC adapter.
- Prevents VTAM security problems when the UNBIND request is used with CICS.
- Supports deletion of LUs automatically on session termination.
- Supports Dynamic LU Naming.
- Supports Inverse DNS Nailing.
- Provides security through SSL Encryption.

TN3270 Server Environments

TN3270 communications in a TCP/IP network consist of the following basic elements:

- TN3270 client—Emulates a 3270 display device for communication with a mainframe application through a TN3270 server over an IP network. The client can support the standard TN3270 functions (as defined by RFC 1576) or the enhanced functionality provided by TN3270E (defined in RFC 2355). TN3270 clients are available on a variety of operating system platforms.

- TN3270 server—Converts the client TN3270 data stream to SNA 3270 and transfers the data to and from the mainframe.
- Mainframe—Provides the application for the TN3270 client and communicates with the TN3270 server using Virtual Telecommunications Access Method (VTAM).

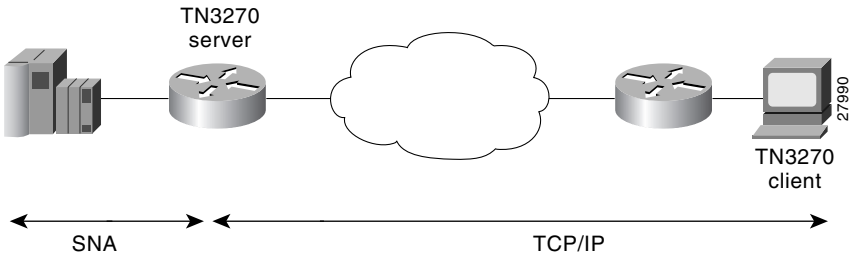
The TN3270 server feature offers an attractive solution when the following conditions need to be supported in an SNA environment:

- Maintaining an IP backbone while providing support for SNA 3270-type clients.
- Offloading mainframe CPU cycles when using a TN3270 host TCP/IP stack with a TN3270 server.
- Providing support for high session density or high transactions per second.

The TN3270 server feature on a CMCC adapter card provides mapping between an SNA 3270 host and a TN3270 client connected to a TCP/IP network as shown in Figure 274. Functionally, it is useful to view the TN3270 server from two different perspectives:

- SNA Functions, page 4
- Telnet Server Functions, page 4

Figure 274 TN3270 Implementation



SNA Functions

From the perspective of an SNA 3270 host connected to the CMCC adapter, the TN3270 server is an SNA device that supports multiple PUs, with each PU supporting up to 255 logical units (LUs). The LU can be Type 1, 2, or 3. The SNA host is unaware of the existence of the TCP/IP extension on the implementation of these LUs.

The LUs implemented by the TN3270 server are dependent LUs. To route these dependent LU sessions to multiple VTAM hosts connected to the TN3270 server in the CMCC adapter card, rather than routing in the VTAM hosts, the TN3270 server implements a SNA session switch with end node (EN) dependent LU requester (DLUR) function. SNA session switching allows you to eliminate SNA subarea routing between hosts of TN3270 traffic by establishing Advanced Peer-to-Peer Networking (APPN) links with the primary LU hosts directly.

Using the DLUR function is optional so that the TN3270 server can be used with VTAM versions prior to version 4.2, which provide no APPN support. In these non-APPN environments, access to multiple hosts is accomplished using direct PU configuration in the TN3270 server.

Telnet Server Functions

From the perspective of a TN3270 client, the TN3270 server is a high-performance Telnet server that supports Telnet connections, negotiation and data format. The server on the CMCC adapter card supports Telnet connection negotiation and data format as specified in RFC 1576 (referred to as *Traditional TN3270*) and RFC 2355 (referred to as *TN3270 Enhancements*).

Unless the TN3270 server uses a Token Ring connection to a front-end processor (FEP), or other LLC connectivity to the mainframe host, it will require CSNA or CMPC support. For more information about configuring CSNA or CMPC support, see the “Configuring CSNA and CMPC” chapter in this publication.

TN3270 Server Architecture

The Cisco TN3270 server can be placed on a channel-attached router or a remote router. If the router is directly connected to the host, the TN3270 server resides on a CIP or CPA that is connected to the mainframe using Enterprise Systems Connection (ESCON) or bus-and-tag channel attachment.

Alternatively, you can use the TN3270 server on a remote router as an intermediate step toward using the CIP or CPA as a direct host connection. In this case, the TN3270 server resides on a router that is connected to the mainframe using a channel connection device, such as the FEP or a CIP or CPA.

The TN3270 server feature is implemented on the following CMCC adapters:

- CIP—Installed in a Cisco 7000 with RSP7000 or 7500 series router. Each CIP has up to two ESCON or two bus-and-tag (parallel) interfaces and a single virtual interface. The TN3270 server is installed on the virtual interface. Therefore, each CIP can have a single TN3270 server.
- CPA—ECPA or PCPA installed in a Cisco 7200 series router. Each CPA combines the function of an ESCON interface and a virtual interface on a single interface. As with the CIP, a single TN3270 server can be installed on each CPA.

Because a router can accommodate more than one CIP or CPA, each router can support multiple TN3270 servers.

Supported PU Types

The TN3270 server supports two types of PUs:

- Direct PUs—Used in subarea SNA
- DLUR PUs—Used with APPN

Direct PUs and DLUR PUs can coexist on the same CIP or CPA. Both types of PUs support either static or dynamic LUs. However, the LU type is defined only in VTAM and is not explicitly defined in the TN3270 server.

Direct PUs

The TN3270 server supports direct PUs when you want to configure a PU entity that has a direct link to a host. Direct PUs are used in non-APPN environments.

The definition of each direct PU within the router requires that you define a local service access point (SAP). Each PU on the TN3270 server must have a unique local/remote media access control (MAC)/SAP quadruple. If you want to connect PUs on the same adapter to the same remote MAC (RMAC) and remote SAP (RSAP), then you must configure each PU with a different link SAP (LSAP).

With direct PUs, the LU names in the TN3270 server do not necessarily match the LU names defined in VTAM. However, there are a couple of ways to accomplish matching LU names for direct PUs:

- LU seed configuration—To ensure that the LU seed configurations in the router and VTAM match for direct PUs, you need to define the value for the **lu-seed** parameter in the **pu** (TN3270) or **pu** (listen-point) command in the router, the same as the LUSEED value in the VTAM PU definition.
- INCLUD0E function available as of VTAM version 4.4—To allow the XCA to provide the LU name in the ACTLU message, use the INCLUD0E function. The TN3270 server then uses the LU name provided by the ACTLU.

DLUR PUs

When the SNA network uses APPN and the TN3270 server can reach multiple hosts, the DLUR function of the TN3270 server is recommended. Note that by using the DLUR function of the TN3270 server, all of the LUs in the server can be defined and owned by a controlling VTAM. When a client requests an application residing on a different VTAM host, the controlling VTAM will issue the request to the target host which will send a BIND directly to the client. All LU-LU data will then flow directly between the target host and the client without needing to go through the controlling VTAM.

DLUR allows the routing of TN3270 LUs to be performed in the CMCC adapter card using SNA session switching to multiple VTAM hosts rather than routing the sessions on the VTAM hosts. This feature is especially important with the multi-CPU CMOS mainframe, which comprises up to 16 CPUs that appear as separate VTAMs.

The implementation of TN3270 server LUs under DLUR also allows the server to learn about the LU names from VTAM in the ACTLU message, which greatly simplifies the configuration to support specifically requestable LUs such as printers.

Supported LU Types

The TN3270 server supports two types of LUs:

- Static LUs—Defined explicitly within VTAM. Allocation of static LUs requires a client to specify the PU and LU name. LU name requests are only supported by TN3270E clients.
- Dynamic LUs—Use the DDDLU feature of VTAM. Allocation of dynamic LUs requires a client to specify only a terminal type. LU name requests to be fulfilled by DDDLUs for PUs configured with the **generic-pool deny** command are supported.

The type of LU that is allocated is defined only in the VTAM switched major node. The TN3270 server does not specify the LU type.

LU Names in the TN3270 Server

Where SNA session switching is configured using DLUR PUs, the TN3270 server learns the LU names (static or dynamic) from VTAM in the ACTLU message. Direct PUs can also learn names from VTAM in the ACTLU message if the INCLUD0E parameter (available in VTAM version 4.4) is used in the switched major node definition.

However, for direct PUs, the TN3270 server can also specify a naming convention that it will use for any dynamic LUs that are allocated. For direct PUs a “seed” name can be configured on the PU in the TN3270 server configuration by using the **lu-seed** argument of the **pu** (TN3270) or **pu** (listen-point) command. The LU seed name defines a prefix for the LU name. The TN3270 server uses the LU seed name in conjunction with the LOCADDR to generate the name by which the TN3270 server recognizes that LU. It is important to note that VTAM also generates LU names using its own LUSEED parameter.

When using the **lu-seed** parameter in the TN3270 server configuration, it is best to use the same naming convention as the host to prevent situations where the LU name that the TN3270 server recognizes differs from the corresponding LU name assigned in VTAM.

Several factors determine how LUs are assigned and named. For more information about the different factors that influence LU naming, see the *TN3270 Design and Implementation Guide* available on Cisco.com.

LU Allocation

This section provides information about the following aspects of LU allocation:

Formation of LU Model Type and Number

VTAM requires a model type and number in the Reply PSID NMVT from the TN3270 server to find an appropriate LU template in the LUGROUP major node. The model type is a four character string and the model number is a two or three character string.

The TN3270 server translates the following formats of terminal type string from a client:

- IBM-<XXXX>-<Y>[-E]: Specifies “XXXX0Y”or “XXXX0YE” in the model type and number field of the Reply PSID NMVT.

Note The “E” in the model string refers to 3270 Extended Datastream. It has no association with the “E” in “TN3270E.”

- IBM-DYNAMIC: Specifies “DYNAMIC” in the model type and number field of the Reply PSID NMVT. The VTAM configuration also must have “DYNAMIC” defined as a template in the LUGROUP.

All other terminal strings that do not match the above syntax examples are forwarded as is to VTAM. For example, a string of “IBM-ZZ..Z,” where “ZZ..Z” does not match the preceding syntax, is forwarded as “ZZ..Z.”

In all cases, the string is translated from ASCII to EBCDIC and truncated at seven characters.

Clients that do not support TN3270E typically require a 3270 datastream on the System Services Control Point (SSCP)-LU flow. Clients that are TN3270E compliant typically use the SNA Character Set (SCS) on the SSCP-LU session. In order to accommodate these two classes of clients, the TN3270 server directs them to different LUGROUP entries at the host. To make this as easy as possible, the SCS requirement is also encoded into the model string sent to the host. Following the previously described terminal type string formats accepted by the server, this additional condition is applied:

If the client has negotiated TN3270E support, the character “S” is overlaid on the fifth character of the string, or appended if the string is less than five characters as shown in Table 18.

Table 18 Examples of Model String Mapping

String from Client (ASCII)	BIND-IMAGE Requested?	String to Host (EBCDIC)
IBM-3278-4	No	327804
IBM-3279-5E	No	327905E
IBM-3279-3-E	Yes	3279S5E
IBM-DYNAMIC	Yes	DYNASIC
ABC	Yes	ABCS
ABCDEFGH	Yes	ABCDSFG

Static LU Allocation

A TN3270E client can request a specific LU name by using the TN3270E command CONNECT as documented in RFC 2355. The name requested must match the name by which the TN3270 server knows the LU and the host must have activated the LU with an ACTLU.

TN3270 clients can also use static LUs if client nailing is configured on the TN3270 server.

Dynamic LU Allocation

Dynamic LU allocation, using VTAM's DDDLU feature, is the most common form of request from TN3270 clients emulating a TN3270 terminal. The user typically requests connection as a particular terminal type and normally is not interested in what LOCADDR or LU name is allocated by the host, as long as a network solicitor logon menu is presented. In fact, only TN3270E clients can request specific LUs by name.

The TN3270 server performs the following functions with this type of session request:

- Forms an EBCDIC string based on the model type and number requested by the client (see the "Formation of LU Model Type and Number" section on page 7 for information about the algorithm used). This string is used as a field in a Reply product set ID (PSID) network management vector transport (NMVT).
- Allocates a LOCADDR from the next available LU in the generic LU pool. This LOCADDR is used in the NMVT.
- Sends the formatted Reply PSID NMVT to VTAM.

To support DDDLU, the PUs used by the TN3270 server have to be defined in VTAM with LUSEED and LUGROUP parameters. When VTAM receives the NMVT it uses the EBCDIC model type and number string to look up an LU template under the LUGROUP. For example, the string "327802E" finds a match in the sample VTAM configuration shown in Figure 278 in the "VTAM Host Configuration Considerations" section on page 19. An ACTLU is sent and a terminal session with the model and type requested by the client is established.

LU name requests to be fulfilled by DDDLUs for PUs configured with the **generic-pool deny** command are supported.

For more information about defining the LUSEED and LUGROUP parameters in VTAM, see the "VTAM Host Configuration Considerations" section on page 19.

Dynamic LU Naming

The Dynamic LU Naming enhancement allows the user to configure named logical units (LUs) from the TN3270 server side. This enhancement allows the TN3270 server to pass an LU name to the Virtual Telecommunications Access Method (VTAM) software running on the mainframe and have VTAM dynamically create an LU with that name. The LU name is then sent to the mainframe as part of subvector 86 in the Reply PSID NMVT power-on frame. The TN3270 client can connect to any of the available TN3270 servers and the selected server can request a specific LU name for the client. In addition, the LU naming conventions have been modified to allow for more flexibility when specifying lu-seed names.

LU Nailing

The TN3270 server allows a client IP address to be mapped or "nailed" to one or more LU local addresses on one or more physical units (PUs) by means of router configuration commands. LU nailing allows you to control the relationship between the TN3270 client and the LU.

Using LU nailing, clients from traditional TN3270 (non-TN3270E) devices can connect to specific LUs, which overcomes a limitation of TN3270 devices that cannot specify a "CONNECT LU." LU nailing is useful for TN3270E clients because it provides central control of your configuration at the router rather than at the client.

The "model matching" feature of Cisco's TN3270 server is designed for efficient use of dynamic LUs. Each TN3270E client specifies a terminal model type at connection. When a non-nailed client connects and does not request a specific LU, the LU allocation algorithm attempts to allocate an LU that operated with that terminal model the last time it was used. If no such model is available, the next choice is an LU that has not been used since the PU was last activated. Failing that, any available LU is used; however, for dynamic LUs only, there is a short delay in connecting the session.

When a client or set of clients is nailed to a set of more than one LU, the same logic applies. If the configured LU nailing maps a screen client to a set of LUs, the LU nailing algorithm attempts to match the client to a previously used LU that was most recently used with the same terminal model type as requested by the client for this connection. If a match is found, then that LU is used. If a match is not found, any LU in the set that is not currently in use is chosen. If there is no available LU in the set, the connection is rejected.

For example, the following LUs are nailed to clients at address 192.195.80.40, and LUs BAGE1004 and BAGE1005, which were connected but are now disconnected.

```
lu    name    client-ip:tcp       nail state    model    frames in out    idle for
1    BAGE1001 192.195.80.40:3822   Y   P-BIND   327904E  4       4       0:22:35
2    BAGE1002 192.195.80.40:3867   Y   ACT/SESS 327904E  8       7       0:21:20
3    BAGE1003 192.195.80.40:3981   Y   ACT/SESS 327803E  13      14      0:10:13
4    BAGE1004 192.195.80.40:3991   Y   ACT/NA   327803E  8       9       0:0:7
5    BAGE1005 192.195.80.40:3997   Y   ACT/NA   327805   8       9       0:7:8
```

If a client at IP address 192.195.80.40 requests a terminal model of type IBM-3278-5, LU BAGE1005 will be selected over BAGE1004.

```
lu    name    client-ip:tcp       nail state    model    frames in out    idle for
1    BAGE1001 192.195.80.40:3822   Y   P-BIND   327904E  4       4       0:23:29
2    BAGE1002 192.195.80.40:3867   Y   ACT/SESS 327904E  8       7       0:22:14
3    BAGE1003 192.195.80.40:3981   Y   ACT/SESS 327803E  13      14      0:11:7
4    BAGE1004 192.195.80.40:3991   Y   ACT/NA   327803E  8       9       0:1:1
5    BAGE1005 192.195.80.40:4052   Y   ACT/SESS 327805   13      14      0:0:16
```

Inverse DNS Nailing

The Inverse DNS Nailing enhancement enables the TN3270 server to nail a pool of LUs to client machine names or to an entire domain. This enhancement allows dynamic IP addressing on the TN3270 client machines. This addressing is used in network design scenarios (for example, a Dynamic Host Configuration Protocol [DHCP] environment) and in individual network configuration scenarios (for example, a machine is moved and needs a new network address).

The Cisco IOS software inverse nailing support uses the DNS in routers to look up the symbolic name associated with a client IP address. The TN3270 server uses this symbolic name to assign a predefined LU pool for the user. This eliminates the need for nailed TN3270 clients to have statically defined IP addresses. If you configure inverse DNS nailing on the TN3270 server, you do not need to modify the DNS nailing statements in the router configuration.

LU Pooling and ASSOCIATE Requests

The TN3270 server enhancements introduced in Cisco IOS Release 12.0(5)T add support for the ASSOCIATE request through LU pooling. The LU pooling feature enables the TN3270 server to identify the relationships between screen and printer LUs.

The LU pool configuration is an option to the LU nailing feature that allows clients to be nailed to LUs. The LU pooling feature allows you to configure clients in the router and nail clients into groups of LUs. These groups of LUs are called clusters. Each cluster is given a unique pool name. An LU pool consists of one or more LU clusters that are related to each other. This allows logically related clients to connect to LUs that have the same logical relationship with the host. A cluster can contain screen LUs and their associated printer LUs. The pool name can be used instead of a device name on a CONNECT request. LU nailing is supported for LU pools.

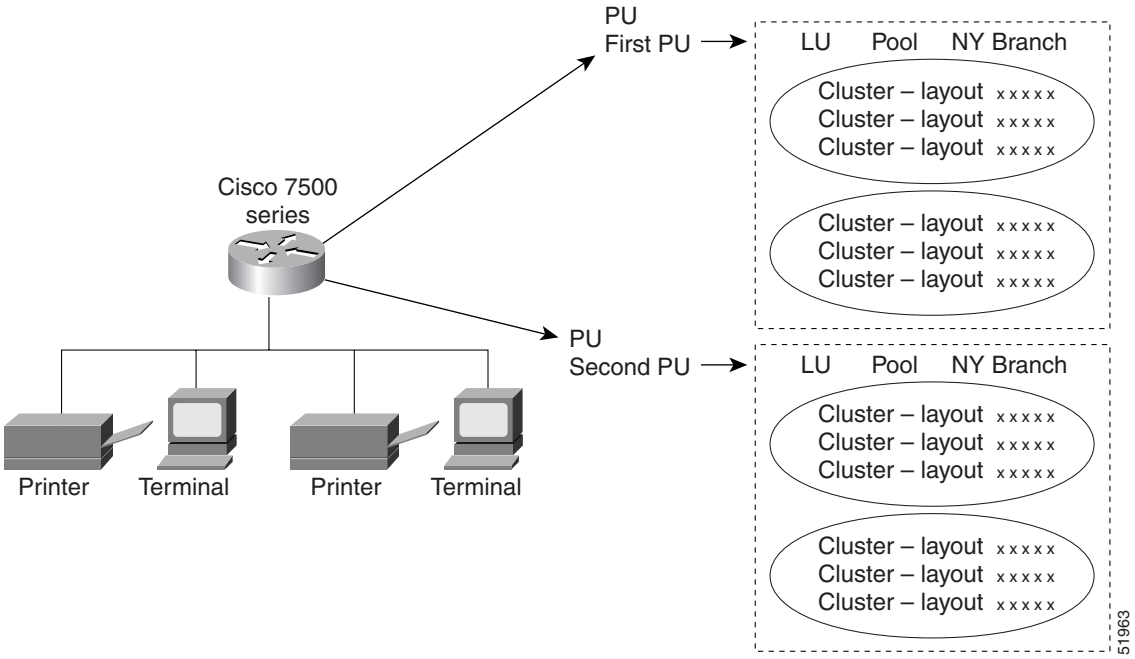
The pool name can be used instead of a device name on a CONNECT request. The pool name must be eight characters or less in length and must comply with VTAM naming rules, which allow the following characters (alphabetic characters are not case sensitive):

- 1st character—Alphabetic (A-Z) and national characters ‘@’, ‘#’, and ‘$’
- 2nd-8th characters—Alphabetic (A-Z), numeric (0-9), and national characters ‘@’, ‘#’, and ‘$’

These naming rules are enforced by the TN3270 server when configuring a pool name and when processing the name received on a CONNECT request from the client. The TN3270 server rejects an invalid name and truncates the name received in the CONNECT request from the client to eight characters or at an invalid character (whichever comes first) when processing the CONNECT request.

Figure 275 provides an overview of clusters configured within PUs.

Figure 275 LU Pooling



Support for the ASSOCIATE request enables you to define a partner printer in the TN3270 server for a given terminal LU pool or single terminal. As a result, the TN3270 server maintains a knowledge of printer and terminal relationships. The client does not need to know the LU name of the partner printer in advance. Typically, a client can request a pool name, a specific LU, or a resource without citing a pool name or LU name.

If the client sends an ASSOCIATE request for a resource name to the TN3270 server, the server provides the client with a resource LU name.

In Figure 276, the client requests an LU from *unixpool* and is granted an LU from the specified pool. The client then initiates a new process by requesting the printer device associated with the given resource LU name.

The client requests a printer LU associated with *termabc* and the server grants the printer LU associated with *termabc*. Based on the configuration in the router that specifies the clusters of printer and screen LUs for pools, the TN3270 server assigns and allows the client to use the printer LU associated with its terminal LU.

Figure 276 Client Request for LU from a Specific Pool and Printer LU Association

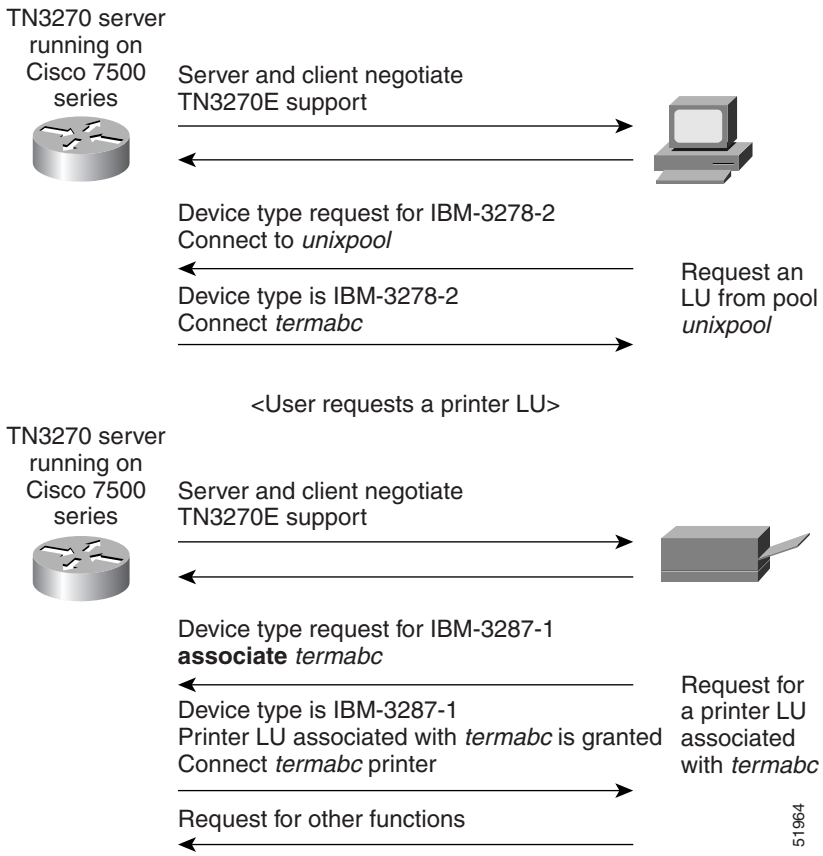
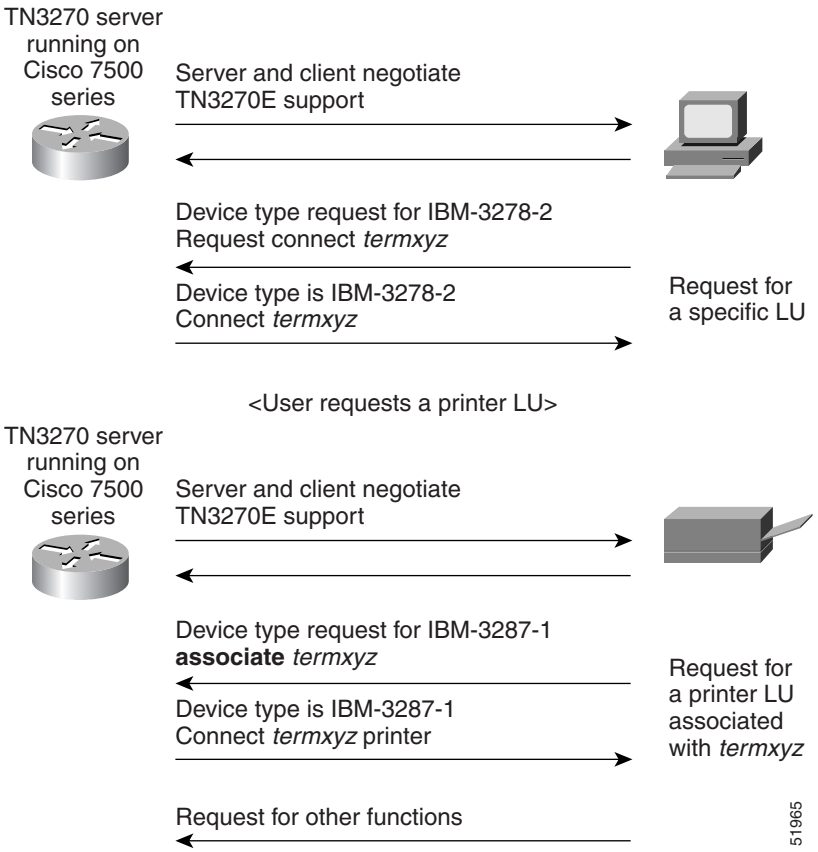


Figure 277 shows the client request for a specific LU *termxyz* and then a request for a printer LU associated with the LU *termxyz*. The TN3270 server grants the screen LU and connects the printer associated with *termxyz*.

Figure 277 Client Request for a Specific LU and Printer LU Association



Pooled LU Allocation

When configured, the pool becomes one of several criteria used by the TN3270 server to assign an LU to a client. When a client requests a connection, the TN3270 server determines the authorized capabilities of the client. For example, the TN3270 server attempts to determine whether LU nailing definitions exist for the client.

When the client criteria is processed, the TN3270 server assigns the first available LU in the group to the client. If an appropriate LU is not found, the TN3270 connection is closed.

Screen and printer LUs for a cluster in a pool are allocated according to the following connection scenarios in the TN3270 server:

- The first client with an IP address that is nailed to a pool connects to the TN3270 server—A cluster is reserved for that client IP address. The first appropriate LU in the cluster that satisfies the client connection request is assigned.
- A client, with the same nailed IP address as a currently connected client, connects to the TN3270 server.
  - Depending on the type of LU requested by the client (screen or printer LU), the first available screen or printer LU within a cluster that is reserved for that nailed IP address is allocated.
  - If there is not an available screen or printer LU in an assigned cluster for the client connection, a new cluster is reserved for clients with that IP address. Then, the first appropriate LU in the cluster that satisfies the client connection request is assigned.

- A client, with a new IP address that is nailed to the same pool as other clients, connects to the TN3270 server—The next available cluster is reserved for that client IP address.
- A client requests a specific pool when connecting to the TN3270 server, but the client IP address is not nailed to the pool—The first available LU in the generic pool is allocated to the client.

For a detailed example of these LU allocation scenarios for a TN3270 server configuration using LU pooling, see the "LU Pooling Configuration Example" section on page 66.

Session Termination

The TN3270 server supports two configuration options that determine how the server responds when a client turns off the device or disconnects:

- LU Termination, page 14
- LU Deletion, page 14

LU Termination

In Cisco IOS Release 12.0(5)T and later, the TN3270 server supports LU termination options for sending either an UNBIND or a TERMSELF RU when a client turns off the device or disconnects from the server.

The **termself** keyword for the **lu termination** command orders termination of all sessions and session requests associated with an LU when a user turns off the device or disconnects from the server. This is an important feature for applications such as IBM's Customer Information Control System (CICS).

If you use an UNBIND request for session termination with CICS, Virtual Telecommunication Access Method (VTAM) security problems can arise. When CICS terminates a session from an UNBIND request, the application may reestablish a previous user's session with a new user, who is now assigned to the same freed LU.

LU Deletion

In Cisco IOS Release 12.0(5)T and later, the TN3270 server adds support for LU deletion options.

The **lu deletion** command specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM when a client disconnects. This command is recommended in host environments running VTAM version 4.4.1. Previous versions of VTAM are not compatible with Network Management Vector Transport (NMVT) REPLY-PSID.

Session Termination Scenarios

Sessions are terminated in the following conditions:

- The client logs off the LU-LU session and the LU is configured to disconnect on UNBIND.
- The client disconnects at the TCP layer.
- The client is idle too long or will not respond to a DO TIMING MARK message.

Any of the above conditions cause the server to do one of the following, depending upon how the **lu termination** command is configured:

- **Unbind** is configured—The TN3270 server sends an UNBIND followed by a NOTIFY (Secondary LU (SLU) DISABLED) message to the host. If the **lu deletion** command is configured to send a REPLY-PSID poweroff request, then the TN3270 server sends the request upon receipt of the NOTIFY response from the host.
- **Termself** is configured—The TN3270 server sends a NOTIFY (SLU DISABLED) to the host. Upon receipt of the NOTIFY response from the host, the TN3270 server sends a TERMSELF request to the host. If the **lu deletion** command is configured to send a REPLY-PSID poweroff request, then the TN3270 server sends the request upon receipt of the TERMSELF response.

Response-Time Collection

Response-time MIB support enables you to capture response-time statistics on the router for either individual sessions and clients or for groups of sessions and clients.

If SNMP is enabled on the router, a network management system (NMS) or users can use well-known and router-configured client group names to obtain response-time statistics. Response-time data collection is always enabled for all in-session clients, excluding printer clients. Table 19 shows the types of client groups that are monitored:

Table 19 Client Group Types and Names

Client Group Type	Description	Client Group Name
Client Subnet	All clients belonging to one or more IP subnets, where the IP subnets and client group name are configured on the router.	User defined
Other	All clients not belonging to an IP subnet configured for a Client Subnet-type group.	CLIENT SUBNET OTHER
Global	All in-session clients.	CLIENT GLOBAL
Application	All clients in session with a specific VTAM APPL ID.	APPL *VTAM-application-name*
Host Link	All clients using a specific host link in use by a PU configured on the router.	DIRECT LINK *pu-name* DLUR LINK *link-name*
Listen Point	All clients connected to a specific listen point configured on the router.	LP *ip-address*: *tcp-port*

The names and IP subnets for the “client subnet” type of response-time group are user-defined. All other client groups are established dynamically by the TN3270 server as clients enter and exit applications. These client groups are named according to the format shown in the column labeled Client Group Name in Table 18.

In Cisco IOS Release 12.2, traps are not generated by the MIB.

Response-time data is collected using the following methods:

- Sliding-Window Average Response Times, page 16
- Response-Time Buckets, page 16

Sliding-Window Average Response Times

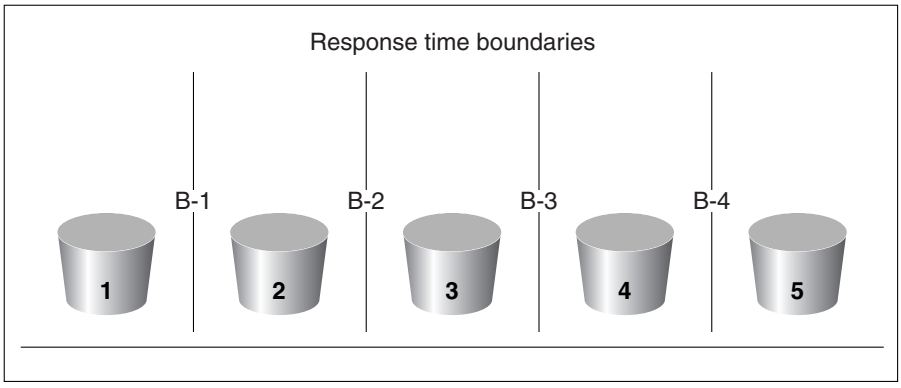
The sliding-window response-time method uses a moving average. It reflects the most recent response time and discounts the old response times. When there is no activity, this method preserves the old response times. The algorithm used for the sliding-window method is similar to the moving-average method. For detailed information about sliding-window average times, refer to the TN3270E-RT-MIB.

Response-Time Buckets

Response-time buckets contain counts of transactions with total response times that fall into a set of specified ranges. Response-time data gathered into a set of five buckets is suitable for verifying service-level agreements or for identifying performance problems through a network management application. The total response times collected in the buckets is governed by whether IP network transit times are included in the totals.

In Figure 278, four bucket boundaries are specified for a response-time collection, which results in five buckets.

Figure 278 Response-Time Boundaries



The first response-time bucket counts transactions with total response times that are less than or equal to boundary 1 (B-1), the second bucket counts transactions with response times greater than B-1 but less than or equal to B-2, and so on. The fifth bucket is unbounded, and it counts all transactions with response times greater than boundary 4.

The four bucket boundaries have default values of 1 second, 2 seconds, 5 seconds, and 10 seconds, respectively.

For a detailed explanation of response-time buckets, refer to the *TN3270E-RT-MIB*.

SSL Encryption Support

The SSL Encryption Support enhancement allows TN3270 clients and servers to negotiate authentication and encryption schemes using the Secure Socket Layer (SSL) technology. The TN3270 server uses SSL version 3.0 to establish secure sessions.

Preparing to Configure the TN3270 Server

Read the following sections to find important information that is useful to know before you configure the TN3270 server:

Hardware and Software Requirements

This section provides the following information about the hardware and software required to use the TN3270 server:

Router Requirements

The Cisco TN3270 server consists of a system image and a microcode image, which are virtually bundled as one combined image.

The following versions of hardware microcode are supported for the CIP and CPA in Cisco IOS Release 12.1:

- CIP hardware microcode—CIP27-2 and later
- CPA hardware microcode—XCPA27-2 and later

The following versions of hardware microcode are supported for the TN3270 Server Connectivity Enhancements feature on the CIP and CPA in Cisco IOS Release 12.1(5)T:

- CIP hardware microcode—CIP28-1 and later
- CPA hardware microcode—XCPA28-1 and later

To enable the TN3270 server feature, you must have a CMCC adapter installed in a Cisco 7000 with RSP7000, Cisco 7200 series router, or a Cisco 7500 series router.

For additional information about what is supported in the various releases of the Cisco IOS software and the CIP microcode, see the information on Cisco.com.

Inverse DNS Nailing

To use inverse DNS Nailing on the TN3270 server, you must specify which DNS servers are required to resolve the TN3270 server client IP addresses. To specify the DNS servers, use the following commands:

- **ip domain-lookup**
- **ip domain-name**
- **ip name-server**

SSL Encryption

To use TN3270 server SSL encryption, you must be running an IOS image with IPSec support. The strength of the SSL encryption support on the TN3270 server is determined by the strength of the IPSec image.

A server digital certificate loaded on the TN3270 router is also required.

Mainframe Requirements

Mainframe hosts using SNA with the TN3270 server must be running VTAM V4R2 or later.

Note You can use VTAM V3R4, but DLUR operation is not supported in V3R4 and proper DDDLU operation may require program temporary fixes (PTFs) to be applied to VTAM.

Dynamic LU Naming

The TN3270 server creates and deletes LUs dynamically on VTAM by sending Reply PSID poweron and Reply PSID poweroff messages when the named LU is connected and disconnected. To properly delete the dynamically created LUs, VTAM requires the following APARS:

- OW41274
- OW41686
- OW40315

You must replace the default exit ISTEXCSD with the VTAM User Exit for TN3270 Name Pushing, which you can download from the IBM website: http://www.ibm.com. This exit causes VTAM to ignore the LUSEED parameter on the PU statement, and instead use the SLU name sent by the router in the subvector 86 when a client connects in. If you do not configure this exit, VTAM ignores the subvector 86 and the specified LU name.

- If you specify the LUSEED operand for the PU definition in VTAM, and the subvector 86 specifies an LU name, the VTAM User Exit for TN3270 Name Pushing ignores the LUSEED operand.
- If you do not specify the LUSEED operand for the PU definition in VTAM, and the subvector 86 is not present, then the VTAM User Exit for TN3270 Name Pushing cannot generate an LU name. VTAM does not log this failure, and the TN3270 server does not receive the ACTLU request. The TN3270 server displays the following message:

```
*Apr 17 12:40:53:%CIP2-3-MSG:slot2 :
%TN3270S-3-NO_DYN_ACTLU_REQ_RCVD
  No ACTLU REQ received on LU JJDL1.6
```

Inverse DNS Nailing

If there are legacy and inverse DNS nailing statements, the inverse DNS nailing statements take precedence. The TN3270 server attempts an inverse DNS lookup before it checks for any legacy nailing configuration.

Cisco strongly recommends that you configure inverse DNS nailing on a PU that does *not* support generic LUs, or on a PU that has the **generic-pool** command configured but also has the **deny** keyword specified.

TN3270 Client Requirements

Based on the RFC standards, the Cisco TN3270 server supports any client that implements the TN3270 or TN3270E protocols.

Design Considerations

The number of sessions that a single TN3270 server can handle is directly related to the number of transactions per second and the amount of memory available to the CIP or CPA. There are other issues to be considered depending upon the environment that you want to support with the TN3270 server.

For comprehensive information about VTAM and router configuration issues and implementing specific TN3270 server scenarios, refer to the *TN3270 Design and Implementation Guide*.

Handling Large Configurations

The maximum size nonvolatile random-access memory (NVRAM) for the Cisco 7000, Cisco 7200, and Cisco 7500 series routers is 128 KB. The maximum number of nailing commands (commands that map IP addresses to LUs) that can be stored in a 128 KB NVRAM is approximately 4000. However, large configurations may contain as many as 10,000 nailing commands.

To maintain a configuration file that exceeds 128 KB there are two alternatives:

- Store the configuration file compressed in NVRAM.
- Store the configuration file in Flash memory (either internal Flash or on a PCMCIA card).

For more information about maintaining configuration files, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide*. For information about router hardware and memory, refer to the hardware configuration guide for your Cisco router series.

Configuring Host Connections

Before configuring the TN3270 server, host connectivity must be configured using one of the following methods:

- Configuring CMPC support
- Configuring CSNA support
- Configuring Token Ring attachment to an FEP

For information about configuring CMPC or CSNA, see the “Configuring CSNA and CMPC” chapter in this publication.

VTAM Host Configuration Considerations

Other non-Cisco implementations of TN3270 support depend on predefined, static pools of LUs to support different terminal types requested by the TN3270 clients. The Cisco TN3270 server implementation on the CMCC adapter removes the static nature of these configurations by using a VTAM release 3.4 feature called DDDLU. DDDLU dynamically requests LUs using the terminal type provided by TN3270 clients. The dynamic request eliminates the need to define any LU configuration in the server to support TN3270 clients emulating a generic TN3270 terminal.

To support DDDLU, the PUs used by the TN3270 server have to be defined in VTAM with LUSEED and LUGROUP parameters, as shown in the following sample configuration:

```
Example VTAM host values defining LUSEED and LUGROUP name parameters:
TN3270PU     PU          .                          *     Defines other PU parameters
                         IDBLK=05D,
                         IDNUM=30001,
                         LUSEED=TN3X1###,           *     Defines the seed component of
                                                          the LU names created by DDDLU
                                                          (e.g. LOCADDR 42 will have the
                                                          name TN3X1042)
                         LUGROUP=AGROUP             *     Defines the LU group name
*
TN3X1100     LU          LOCADDR=100,               *     Defines a terminal which
                         MODETAB=AMODETAB                 requires a specific LU name
*
TN3X1101     LU          LOCADDR=101,               *     Defines a printer which requires
                         DLOGMODE=M3287CS                 a specific LU name


Example VTAM host values defining LUGROUPname, AGROUP:
AGROUP       LUGROUP                                *     Defines LU group to support
                                                          various terminal types
327802E      LU          USSTAB=USSXXX,             *     Defines template to support IBM
                         LOGAPPL=TPXP001,                 3278 terminal model 2 with
                         DLOGMOD=SNX32702,                Extended Data Stream. Note that
                         SSCPFM=USS3270                   the USS messages in USSXXX
                                                          should be in 3270 datastream.
3278S2E      LU          USSTAB=USSYYY,             *     Defines template to support IBM
                         LOGAPPL=TPXP001,                 3278 terminal model 2 with
                         DLOGMOD=SNX32702,                Extended Data Stream, for
                         SSCPFM=USSSCS                    TN3270E clients requesting
                                                          BIND-IMAGE.
327805       LU          USSTAB=USSXXX,             *     Defines template to support IBM
                         LOGAPPL=TPXP001,                 3279 terminal model 5
                         DLOGMOD=D4C32785,
                         SSCPFM=USS3270
@            LU          USSTAB=USSXXX,                   Defines the default template to
                         LOGAPPL=TPXP001,                 match any other terminal types
                         DLOGMOD=D4A32772,
                         SSCPFM=USS3270
```

With the configuration shown above defined in the host, the ACTPU sent by VTAM for the PU TN3270PU will have the “Unsolicited NMVT Support” set in the SSCP capabilities control vector. This allows the PU to dynamically allocate LUs by sending network management vector transport (NMVT) with a “Reply Product Set ID” control vector.

After the TN3270 server sends a positive response to the ACTPU, it will wait for VTAM to send ACTLUs for all specifically defined LUs. In the sample configuration shown in Figure 278, ACTLUs will be sent for TN3X1100 and TN3X1101. The server sends a positive response and sets SLU DISABLED. The LOCADDRs of the TN3X1100 and TN3X1101 LUs are put into the specific LU cache and reserved for specific LU name requests only.

To allow sufficient time for the VTAM host to send all the ACTLUs, a 30-second timer is started and restarted when an ACTLU is received. When the timer expires it is assumed that all ACTLUs defined in VTAM for the PU have been sent. All LUs that have not been activated are available in a generic LU pool to be used for DDDLU unless they have been reserved by the configuration using the **generic-pool deny** TN3270 configuration command.

After the VTAM activation, the server can support session requests from clients using dynamic or specific LU allocation.

For more information about DDDLU in VTAM, refer to the VTAM operating system manuals for your host system under the descriptions for LUGROUP.

Note If your host computer is customized for a character set other than U.S. English EBCDIC, you might need to code some VTAM configuration tables differently than indicated in the examples provided by Cisco.

Some VTAM configurations include the number sign (#) and at symbol (@). In the U.S. English EBCDIC character set, these characters are stored as the hexadecimal values 7B and 7C, respectively. VTAM will look for those hexadecimal values when processing the configuration file.

The characters used to enter these values are different in other EBCDIC National Language character sets. Table 20 lists the languages that have different characters for the 7B and 7C hexadecimal values and the corresponding symbols used to enter the characters.

For example, a parameter with a value of TN3X1### would have a value of TN3X1£££ for the French National Language character set.

Table 20 International Character Sets for Hexadecimal Values

	Hexadecimal Value			
	7B		7C	
Language	**Symbol**	**Description**	**Symbol**	**Description**
German	#	Number sign	§	Section symbol
German (alternate)	Ä	A-dieresis	Ö	O-dieresis
Belgian	#	Number sign	à	a-grave
Brazilian	Õ	O-tilde	Ã	A-tilde
Danish/Norwegian	Æ	AE-ligature	Ø	O-slash
English (U.S./UK)	#	Number sign	@	At symbol
Finnish/Swedish	Ä	A-dieresis	Ö	O-dieresis
French	£	Pound sterling	à	a-grave
Greek	£	Pound sterling	§	Section symbol
Icelandic	#	Number sign	D	Uppercase eth
Italian	£	Pound sterling	§	Section symbol
Portuguese	Õ	O-tilde	Ã	A-tilde
Spanish	Ñ	N-tilde	@	At symbol
Turkish	Ö	O-dieresis	S	S-cedilla

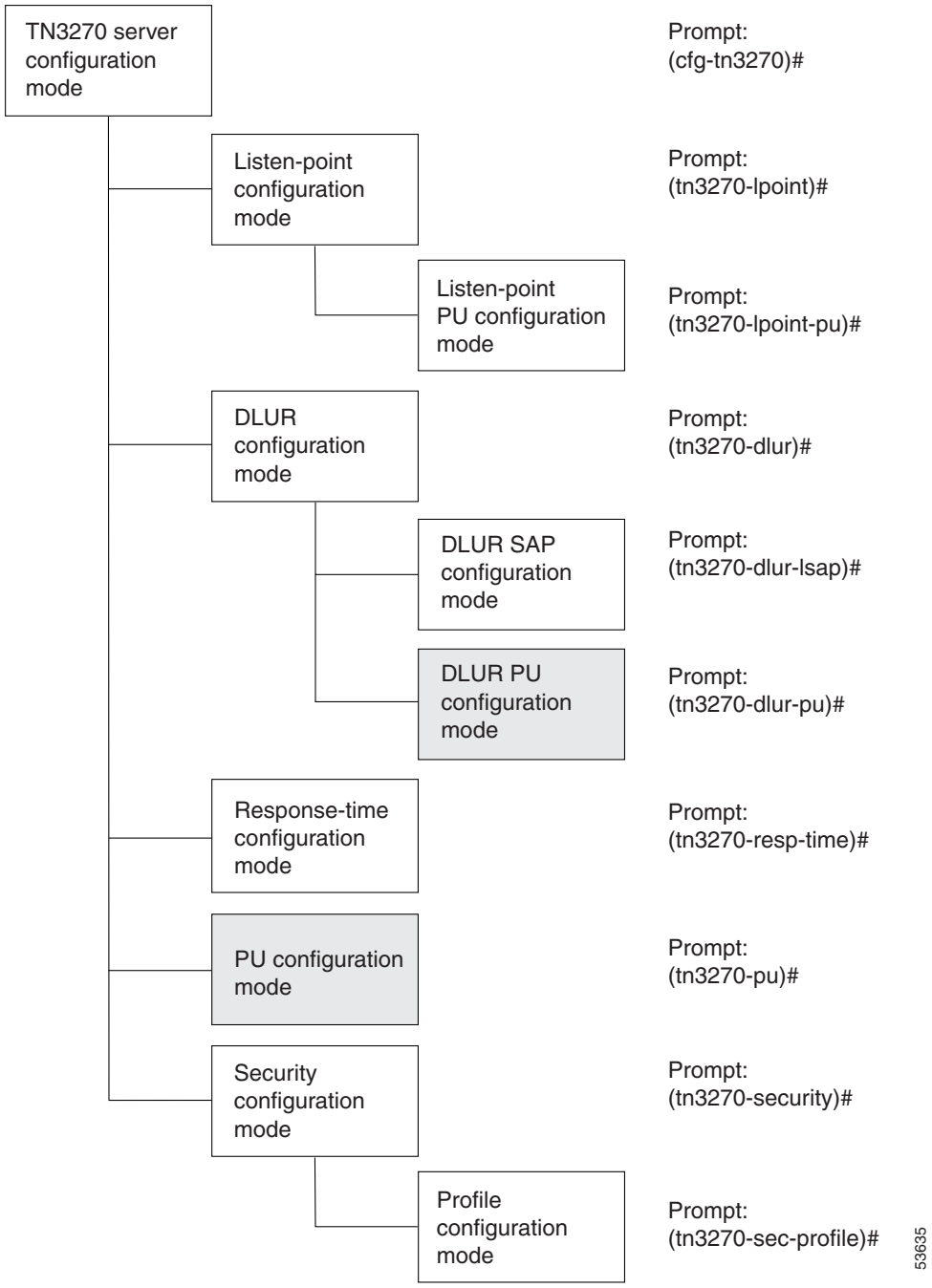
TN3270 Server Configuration Modes

Figure 279 shows the TN3270 configuration modes that are supported in Cisco IOS Release 12.2 and which are described in the following sections of this topic:

The TN3270 server can be configured only on the virtual interface of a CMCC adapter. Some configuration commands create entities on the CMCC adapter. For most of these commands, the command changes to the mode associated with that entity (for example, a PU).

When preparing to configure the TN3270 server it is important to understand how to access and move between these different configuration modes. See the "Moving Between Configuration Modes" section on page 26 for more information.

Figure 279 TN3270 Configuration Modes



Note The DLUR, DLUR SAP, DLUR PU and PU configuration modes existed in Cisco IOS Release 12.0(5)T and earlier. DLUR PU and PU configuration modes (shown in the shaded boxes) are legacy configuration modes, whose functions can be replaced by the listen-point configuration modes in Cisco IOS Release 12.0(5)T and later. For more information about the relationship of these legacy configuration modes to the new listen-point configuration modes, see the "Configuring the TN3270 Server with LU Pooling" section on page 42.

TN3270 Server Configuration Mode

From interface configuration mode, the following **tn3270-server** command puts you in TN3270 server configuration mode:

```
router(config-if)# tn3270-server
```

The following prompt appears:

```
(cfg-tn3270)#
```

Note For the CIP, enter interface configuration mode from the virtual channel interface using port 2; For the CPA, enter interface configuration mode from the physical channel interface using port 0.

Listen-Point Configuration Mode

From the TN3270 server configuration mode, the following **listen-point** command puts you in listen-point configuration mode:

```
router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]
```

The following prompt appears:

```
(tn3270-lpoint)#
```

Listen-Point PU Configuration Mode

From listen-point configuration mode, you can create direct PUs and DLUR PUs:

- From the listen-point configuration mode, the following **pu** (listen-point) command creates a new direct PU:

  ```
  router#(tn3270-lpoint)# pu pu-name idblk-idnum type adapno lsap [rmac rmac] [rsap
  rsap] [lu-seed lu-name-stem]
  ```

  The **pu** (listen-point) command puts you in listen-point PU configuration mode and the following prompt appears:

  ```
  (tn3270-lpoint-pu)#
  ```

- From listen-point configuration mode, the following **pu dlur** command creates a new PU for DLUR:

  ```
  router#(tn3270-lpoint)# pu pu-name idblk-idnum dlur
  ```

  The **pu dlur** command puts you in the listen-point PU configuration mode and the following prompt appears:

  ```
  (tn3270-lpoint-pu)#
  ```

DLUR Configuration Mode

From TN3270 server configuration mode, the following **dlur** command puts you in DLUR configuration mode:

```
router(cfg-tn3270)# dlur fq-cpname fq-dlusname
```

The following prompt appears:

```
(tn3270-dlur)#
```

DLUR PU Configuration Mode

Note DLUR PU configuration mode is a legacy configuration mode whose function to define DLUR PUs can be replaced by using the listen-point configuration modes in Cisco IOS Release 12.0(5)T and later. When you define listen-point configurations, you can create DLUR PUs within listen-point PU configuration mode using the **pu dlur** command instead.

From DLUR configuration mode, the following **pu** (DLUR) command creates a new PU for DLUR:

```
router(tn3270-dlur)# pu pu-name idblk-idnum ip-address
```

The **pu** (DLUR) command puts you in the DLUR PU configuration mode and the following prompt appears:

```
(tn3270-dlur-pu)#
```

DLUR SAP Configuration Mode

From DLUR server configuration mode, the following **lsap** command puts you in DLUR SAP configuration mode:

```
router(tn3270-dlur)# lsap type adapno [lsap]
```

The following prompt appears:

```
(tn3270-dlur-lsap)#
```

Response-Time Configuration Mode

From TN3270 server configuration mode, the following **response-time group** command puts you in response-time configuration mode:

```
router(cfg-tn3270)# response-time group name [bucket boundaries t1 t2 t3 t4...][multiplier
m]
```

The following prompt appears:

```
(tn3270-resp-time)#
```

PU Configuration Mode

Note PU configuration mode is a legacy configuration mode whose function to define direct PUs can be replaced by using the listen-point configuration modes in Cisco IOS Release 12.0(5)T and later. When you define listen-point configurations, you can create direct PUs within listen-point PU configuration mode using the **pu** (listen-point) command instead.

From TN3270 server configuration mode, the following **pu** (TN3270) command creates a new direct PU:

```
router(cfg-tn3270)# pu pu-name idblk-idnum ip-address type adapno lsap [rmac rmac] [rsap
rsap] [lu-seed lu-name-stem]
```

The **pu** (TN3270) command puts you in PU configuration mode and the following prompt appears:

```
(tn3270-pu)#
```

Security Configuration Mode

From the TN3270 server configuration mode, the following **security** command puts you in security configuration mode:

```
router(cfg-tn3270)# security
```

The following prompt appears:

```
(tn3270-security)#
```

Profile Configuration Mode

From security configuration mode, the following **profile** command puts you in profile configuration mode:

```
router(cfg-tn3270)# profile profilename {ssl | none}
```

The following prompt appears:

```
(tn3270-sec-profile)#
```

Moving Between Configuration Modes

In general, the parameters within a configuration mode can be grouped into two categories:

- Parameters to identify the specific instance of the entity (for example, a PU name).
- Parameters to set operating options.

To return to a mode later in the configuration process, use the same configuration command but specify only the first set of identification parameters. The following examples show how to create, access, and remove different TN3270 entities in their associated configuration modes.

Working with a Listen-Point Direct PU

The following example shows how to create, access, and remove a listen-point PU entity:

1. To create a listen-point direct PU entity called PU1 and enter listen-point PU configuration mode from listen-point configuration mode, use the **pu** (listen-point) command as shown in the following example:

   ```
   router(tn3270-lpoint)# pu PU1 94201231 tok 1 10
   ```

2. To return later to the listen-point PU configuration mode for the PU1 entity, use the same **pu** (listen-point) command without the "94201231 tok 1 10" parameters from listen-point configuration mode:

   ```
   router(tn3270-lpoint)# pu PU1
   ```

3. To remove the listen-point PU entity called PU1, use the same command with the **no** keyword:

   ```
   router(tn3270-lpoint)# no pu PU1
   ```

Working with a Listen-Point DLUR PU

The following example shows how to create, access, and remove a listen-point DLUR PU entity:

1. To create a listen-point DLUR PU entity called PU2 and enter listen-point PU configuration mode from listen-point configuration mode, use the **pu dlur** command as shown in the following example:

   ```
   router(tn3270-lpoint)# pu PU2 017ABCDE dlur
   ```

2. To return later to the listen-point PU configuration mode for the PU2 entity, use the same **pu dlur** command without the "017ABCDE dlur" parameters from listen-point configuration mode:

```
router(tn3270-lpoint)# pu PU2
```

3. To remove the listen-point PU entity called PU2, use the same command with the **no** keyword:

```
router(tn3270-lpoint)# no pu PU2
```

Working with a DLUR Entity

The following example shows how to create, access, and remove a DLUR entity:

1. To create a DLUR entity with a control point name NETA.RTR1 and enter DLUR configuration mode from TN3270 server configuration mode, use the **dlur** command as shown in the following example:

```
router(cfg-tn3270)# dlur NETA.RTR1 NETA.HOST
```

2. To return later to the DLUR configuration mode for the NETA.RTR1 entity, use the same **dlur** command without the "NETA.RTR1 and NETA.HOST" parameters from TN3270 server configuration mode:

```
router(cfg-tn3270)# dlur
```

3. To remove the NETA.RTR1 DLUR entity, use the same **dlur** command with the **no** keyword:

```
router(cfg-tn3270)# no dlur
```

Working with a DLUR LSAP Entity

The following example shows how to create, access, and remove a DLUR LSAP entity:

1. To create a DLUR LSAP entity and enter DLUR SAP configuration mode from DLUR mode, type the following command:

```
router(tn3270-dlur)#lsap token-adapter 1 84
```

2. To return later to the DLUR SAP configuration mode on the same entity, use the same **lsap** command without the "84" parameter from TN3270 DLUR mode:

```
router(tn3270-dlur)#lsap token-adapter 1
```

3. To remove the DLUR LSAP entity, use the same identification parameters with the **no** keyword:

```
router(tn3270-dlur)#no lsap token-adapter 1
```

Configuring the TN3270 Server

This section provides information about configuring and verifying the TN3270 server. It describes how to configure the commands that are applicable in multiple configuration modes, and how to configure the many options that are available in the TN3270 server.

This section also describes the tasks to configure the TN3270 server in certain environments, and references the configuration options that are available there. Older TN3270 server configurations that are still supported but are replaced by newer methods of configuration are discussed in the legacy configuration topic.

Finally, this section includes a basic procedure for verifying the TN3270 server configuration.

This section includes the following topics:

See the "TN3270 Server Configuration Examples" section on page 64 for examples.

Configuring TN3270 Siftdown Commands

There are many siftdown commands supported by the TN3270 server in multiple configuration modes. Values that you enter for a siftdown command in a subsequent configuration mode might override the values that you have entered for the same command (for the applicable PU only) in a previous configuration mode as shown in the hierarchy in Figure 279.

Consider the following example in which the **keepalive** (TN3270) command is configured in more than one command mode:

```
tn3270-server
keepalive 300
listen-point 10.10.10.1 tcp-port 40
  pu PU1 94223456 tok 1 08
    keepalive 10 send timing-mark 5
  pu PU2 94223457 tok 2 12
```

In this example the **keepalive** (TN3270) command is first configured in TN3270 server configuration mode, which applies to all PUs supported by the TN3270 server. The **keepalive** command is specified again under the listen-point PU configuration mode for PU1, which overrides the previously specified **keepalive** 300 value, for PU1 only. PU2 continues to use the value of the **keepalive** command in the TN3270 server configuration level.

Table 21 provides a list of the TN3270 siftdown commands and the associated configuration modes in which they are supported. An X in the column indicates that the command is supported. A "–" indicates that the command is not supported.

Table 21 Supported Configuration Modes for TN3270 Siftdown Commands

Siftdown Command	TN3270 Server (cfg-tn3270)#	Listen-Point (tn3270-lpoint)#	Listen-Point PU (tn3270-lpoint-PU)#	DLUR PU (tn3270-dlur-pu)	PU (tn3270-pu)#
generic-pool	X	X	X	X	X
idle-time	X	X	X	X	X
ip precedence	X	X	–	X	X
ip tos	X	X	–	X	X
keepalive	X	X	X	X	X
lu deletion	X	X	X	X	X
lu termination	X	X	X	X	X
tcp-port	X	–	–	X	X
unbind-action	X	X	X	X	X

Note You cannot configure the siftdown commands shown in Table 21 while in DLUR, DLUR SAP, or response-time configuration modes for the TN3270 server.

The siftdown commands apply to the corresponding PUs, according to the configuration mode in which they are entered:

- TN3270 server configuration—The siftdown command at this level applies to all PUs supported by the TN3270 server.
- Listen-point configuration—The siftdown command at this level applies to all PUs defined at the listen point.
- Listen-point PU configuration—The siftdown command at this level applies to only the specified PU.
- PU configuration—The siftdown command at this level applies only to the specified PU.

The **no** form of a siftdown command typically inherits the value from the previously configured siftdown value from the entity above it according to the configuration mode hierarchy shown in Figure 279, or it returns to the default value.

Configuring the TN3270 Server Options

The TN3270 server supports many options, some of which are available in multiple configuration modes. The topics in this section explain background information about the TN3270 server options including why an option is useful and how you can configure it. The configuration procedures that are provided later in this chapter also indicate where the options are available in the configuration task list.

This section describes how to configure the following options for the TN3270 server:

Most of these options are available in multiple command modes and are called “siftdown” commands. For more information about how siftdown commands work, see the “Configuring TN3270 Siftdown Commands” section on page 28.

Refer to the “TN3270 Server Commands” chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2) for additional information about the commands described in this section and chapter.

Configuring a Generic Pool of LUs

Configuring a generic pool of LUs in the TN3270 server specifies that “leftover” LUs from a pool of dynamic LUs are available to TN3270 sessions that do not request a specific LU or LU pool through TN3270E. All LUs in a generic pool are DDDLU capable.

A leftover LU is an inactive LU from a pool of dynamic LUs, which are defined in the switched major node in VTAM using the LU-SEED parameter and the LUGROUP parameter. A leftover LU is defined as an LU where all of the following conditions are true:

- The SSCP did not send an ACTLU during PU start-up.
- The PU controlling the LU is capable of carrying product set ID (PSID) vectors on NMVT messages, thus allowing DDDLU operation for that LU.

The default behavior is to permit a generic pool of LUs in the TN3270 server and allow leftover LUs to be used for dynamic connections. You might deny the use of the generic pool for security reasons.

To configure a generic pool of LUs for the TN3270 server, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# generic-pool {permit | deny}`	(Optional) Specifies whether leftover LUs can be used from a generic LU pool. The available options for this command are: • **permit**—Specifies that leftover LUs can be used by clients that request a generic session. Inactive LUs are immediately available for dynamic connections. This is the default. • **deny**—Specifies that the TN3270 server does not allow any further dynamic connections of any LUs on the PU. That is, only static LUs are supported.

The **generic-pool** command takes effect immediately for all upcoming connections, but existing sessions are unaffected. Once the existing sessions are terminated, then future connections will abide by the latest generic pool configuration for that PU. Use the **no** form of this command to selectively remove the permit or deny condition of generic pool use for the corresponding PU and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **generic-pool** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring Idle-Time

The idle time option in the TN3270 server specifies the allowable duration of inactivity in the client-server session before the TN3270 server disconnects an LU.

To prevent an LU session from being disconnected due to inactivity, specify an idle time value of 0 seconds. Note that TIMING-MARKS generated by the TN3270 server keepalive function are not considered "activity" on the client connection.

Note There are two TN3270 server options that can affect when a session is disconnected—idle time and keepalive. These two options operate independently of each other and both can be used to clean up partially disconnected sessions. Whichever option first detects that a session is eligible for disconnect immediately causes the TN3270 server to disconnect that session. If you are specifying both the idle time and keepalive options, then you might consider how the values for these options determine when client sessions are disconnected to achieve the response that you want.

To configure the allowable amount of idle time before the TN3270 server disconnects an LU, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# idle-time seconds`	(Optional) Specifies the number of seconds of inactivity before the TN3270 server disconnects an LU.

The default behavior in TN3270 server configuration mode is that the session is never disconnected (or, a value of 0). The default value in other configuration modes is the value currently configured for that PU in a previously supported mode. Use the **no** form of this command to cancel the idle time period and return to the default for the corresponding PU.

The **idle-time** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring IP Precedence

Configuring the IP precedence option in the TN3270 server allows you to assign different priority levels to IP traffic on a PU in the TN3270 server. IP precedence values are used with the weighted fair queueing (WFQ) or priority queueing features on a Cisco router to allow you to prioritize traffic. IP precedence and IP ToS values are used together to manage network traffic priorities.

The TN3270 server allows you to specify different IP precedence values for screen and printer clients because the communication requirements for each type of client is different. Screen clients are characterized by interactive communication which normally demands a higher priority of data transfer than printers. Printers are characterized by bulk data transfer where priority of sending the data is not as high.

To configure the traffic priority for screen and printer clients in the TN3270 server, use the following command in TN3270 server, listen-point, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# ip precedence {screen | printer} value`	(Optional) Specifies the precedence level (from 0 to 7) for IP traffic in the TN3270 server. The default value is 0.

Use the **no** form of this command to remove the screen or printer precedence value for the corresponding PU and return to the previously configured siftdown value applicable to the PU, or to the default value. However, you can enter new or different values for IP precedence without first using the **no** form of the command.

The **ip precedence** command in the TN3270 server is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring IP ToS

Configuring the IP ToS option in the TN3270 server allows you to assign different levels of service to traffic on a PU in the TN3270 server. IP ToS values are used with the WFQ and NetFlow switching features on a Cisco router. The Open Shortest Path First (OSPF) protocol can also discriminate between different routes based on IP ToS values. IP ToS and IP precedence values are used together to manage network traffic priorities.

The TN3270 server allows you to specify different IP ToS values for screen and printer clients because the communication requirements for each type of client is different. Screen clients are characterized by interactive communication which normally demands a higher priority of data transfer than printers. Printers are characterized by bulk data transfer where priority of sending the data is not as high.

To configure the level of service for screen and printer clients in the TN3270 server, use the following command in TN3270 server, listen-point, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# ip tos {screen | printer} value`	(Optional) Specifies a type of service level (from 0 to 15) for IP traffic in the TN3270 server.

Use the **no** form of this command to remove the screen or printer ToS value for the corresponding PU and return to the previously configured siftdown value applicable to the PU, or to the default value. However, you can enter new or different values for IP ToS without first using the **no** form of the command.

The **ip tos** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring Keepalive

The keepalive options for the TN3270 server allow you to monitor the availability of a TN3270 client session by sending timing marks or Telnet no operation (**nop**) commands. You can configure the frequency and the type of keepalive that the TN3270 server sends to a client and when the TN3270 server determines that a client is inactive.

When you configure the **keepalive** command to send Telnet **nop** commands, no response is required by the client. If you specify only the keepalive interval, then the TN3270 server sends timing marks.

The default behavior of the TN3270 server is to send timing marks every 30 minutes if there is no other traffic flowing between the TN3270 client and server. The TN3270 server disconnects a session if the client does not respond within 30 seconds.

The **keepalive** command affects currently active and future TN3270 sessions. For example, reducing the keepalive interval for timing marks to a smaller nonzero value causes an immediate burst of DO TIMING-MARKS on those sessions that have been inactive for a period of time greater than the new, smaller value.

Note There are two TN3270 server options that can affect when a session is disconnected—idle time and keepalive. These two options operate independently of each other and both can be used to clean up partially disconnected sessions. Whichever option first detects that a session is eligible for disconnect immediately causes the TN3270 server to disconnect that session. If you are specifying both the idle time and keepalive options, then you might consider how the values for these options determine when client sessions are disconnected to achieve the response that you want.

To configure the keepalive options for the TN3270 server, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# keepalive seconds [send {nop | timing-mark [max-response-time]}]`	(Optional) Specifies the number of seconds (from 0 to 65535) of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. A value of 0 means that no keepalive signals are sent. The default interval is 1800 seconds (30 minutes). The following options are available: • **send nop**—Sends the Telnet command for no operation to the TN3270 client to verify the physical connection. • **send timing-mark** [*max-response-time*]—Sends timing marks to verify the status of the client session and specifies the number of seconds (from 0 to 32767) within which the TN3270 server expects a response. The default maximum response time is 30 seconds if the keepalive interval is greater than or equal to 30 seconds. If the value of the keepalive interval is less than 30 seconds, then the default *max-response-time* is the value of the interval. The value of *max-response-time* should be less than or equal to the interval.

Use the **no** form of the command to cancel the current keepalive period and type and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **keepalive** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring LU Allocation and LU Nailing

With the addition of the LU pooling and listen-point configuration methods in Cisco IOS Release 12.0(5)T, the TN3270 server supports multiple methods of allocating LUs and assigning or "nailing" those LUs to a particular client or group of clients.

The TN3270 server supports nailing individual clients to a specific LU and nailing clients to pools. The individual nailing method is useful when a particular client must use a specific LU. Nailing clients to pools is useful when a client needs to have one of a group of LUs associated with a particular PU. For more information about these methods of LU nailing, see the "Methods of LU Nailing" section on page 54.

LU pooling configuration methods using listen points provides an efficient means of configuring clusters of screens and printer LUs into pools, and allocating LOCADDRs. Then, multiple clients can be assigned or "nailed" to those pools to be given access to those LUs.

Note You cannot specify the same LOCADDR in both an individual LU nailing statement and in a pool. The CMCC adapter does not allow a LOCADDR to be allocated multiple times, so the LU allocations in the TN3270 server must not overlap.

Nailing Clients to Specific LUs

To nail a client to a specific LU use the following command in PU configuration mode or listen-point PU configuration mode:

Command	Purpose
`Router# client [printer] ip ip-address [mask] lu first-locaddr [last-locaddr]`	(Optional) Allocates a specific LU or range of LUs to a client located at the IP address or subnet.

Nailing Clients to Pools

To nail a client to a pool of LUs use the following command in listen-point configuration mode:

Command	Purpose
`Router(tn3270-lpoint)# client ip ip-address [mask] pool poolname`	(Optional) Nails a client located at the IP address or subnet to a pool.

Allocating LUs to Pools

To allocate LUs to a pool use the following command in listen-point PU configuration mode:

Command	Purpose
`Router(tn3270-lpoint-pu)# allocate lu lu-address pool poolname clusters count`	(Optional) Assigns LUs to the pool beginning with the LOCADDR specified by *lu-address* for a total of *count* LUs.

Configuring LU Deletion

The LU deletion options for the TN3270 server specify whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects. The LU deletion command is useful to prevent screen LUs from attaching to an LU that was used by a previous session that designates an incompatible screen size for the current LU.

The default behavior of the TN3270 server is to never delete LUs upon disconnect. This option is useful when you only have screen LUs and they all use the same screen size.

To configure the LU deletion options for the TN3270 server, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies when the TN3270 server sends a REPLY-PSID poweroff request for an LU upon disconnect. The following options are available: • **always**—Specifies deletion of all dynamic LUs upon disconnect. • **normal**—Specifies deletion of only screen LUs upon disconnect. • **non-generic**—Specifies deletion of specified LUs. (Available when VTAM supports deletion of specifically-named LUs. Not available as of VTAM version 4.4.1.) • **never**—Specifies that LUs are never deleted upon disconnect. This is the default.

Use the **no** form of the command to remove LU deletion from the current configuration scope and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **lu deletion** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

For additional information about how sessions are terminated, see the "Session Termination" section on page 14.

Configuring LU Termination

The LU termination options for the TN3270 server specify the type of RU sent by the TN3270 server upon LU disconnect. The default behavior of the TN3270 server is to send an UNBIND request to the application to terminate the session.

With some applications (such as CICS), VTAM security problems can arise from an UNBIND request. In some cases the application might reestablish a previous user's session with a new user, who is now assigned to the same freed LU. To prevent this you can configure the TN3270 server to send a TERMSELF RU.

Use the **termself** keyword of the **lu termination** command when you want to be sure that the application terminates the session when the LU disconnects.

To configure the LU termination options for the TN3270 server, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router# lu termination {termself | unbind}`	(Optional) Specifies the type of RU sent by the TN3270 server when a client turns off the device or disconnects. The following options are available: • **termself**—Orders termination of all sessions and session requests associated with an LU upon disconnect. • **unbind**—Requests termination of the session by the application upon LU disconnect. This is the default.

Use the **no** form of the command to remove LU termination from the current configuration scope and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **lu termination** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

For additional information about how sessions are terminated, see the "Session Termination" section on page 14.

Configuring the Maximum Number of Sessions Supported by the Server

Configuring the maximum number of LU control blocks on the TN3270 server determines the limit on the number of sessions that the TN3270 server can support on the CMCC adapter. The practical limit (within the allowable range for the option) is determined in part by your licensing structure for the CMCC and on your hardware and usage characteristics.

Each control block uses about 1 KB of memory, with a possible 2 KB per LU additionally required for data during session activity. The TN3270 server attempts to allocate one LU control block for each LU activated by the host. For DDDLU, the control block is allocated when the client requests the LU, in anticipation of an ACTLU from the SSCP host.

By limiting the number of LU control blocks allocated, you can limit how much memory is used for the TN3270 server and be sure that memory is available to support other CMCC functions.

To configure the maximum number of LUs allowed for the TN3270 server, use the following command in TN3270 server configuration mode:

Command	Purpose
`Router(cfg-tn3270)# maximum-lus number`	(Optional) Specifies the maximum number (between 0 and 32000) of LU control blocks allowed for the TN3270 server. The default is 2100.

Use the **no** form of the command to restore the default value. Although you can change the value of the **maximum-lus** command at any time, you must deactivate the PU (DACTPU) or use the **no pu** command to free allocated control blocks if you reduce the maximum number below the current number of allowable LU control blocks.

Configuring the Maximum Number of Sessions That Can be Obtained by a Single Client

Configuring the maximum number of LU sessions for a TN3270 client limits the number of LU sessions that a client at a specified IP address or IP subnet can establish with the TN3270 server. Establishing this limit prevents a single workstation from using all of the available resources on the TN3270 server. If you configure LU pools and maximum LU sessions, the maximum LU session value limits the number of LOCADDRs that a client can connect to across all pools to which the client belongs.

If you do not configure the maximum number of LU sessions, the default configuration specifies no limit on the number of concurrent sessions from one client IP address.

To configure the maximum number of LU sessions allowed for a TN3270 client, use the following command in TN3270 server configuration mode:

Command	Purpose
`Router(cfg-tn3270)# client [ip [ip-mask]] lu maximum number`	(Optional) Specifies the maximum number of LU sessions (between 0 and 65535) for each client IP address or IP subnet address.

Use the **no** form of the command to remove a single LU limit associated with a particular IP address, or to restore a default value of 65535.

Note There is no relationship between the **allocate lu** command and the **client lu maximum** command. The **allocate lu** command assigns named LOCADDRs to a pool. More than one TN3270 client can access pools and there is no relationship between the number of LUs assigned to a pool and the maximum number of LUs that one client can use.

Configuring the TCP Port

Configuring the TCP port option allows you to override the default TCP port setting of 23, which is the Internet Engineering Task Force (IETF) standard. The value of 65535 is reserved by the TN3270 server.

There are two ways that you can configure the TCP port:

- Using TN3270 server or PU configuration modes for the PU. This is the only method supported in legacy configurations, prior to Cisco IOS Release 12.0(5)T.
- In Cisco IOS Release 12.0(5)T and later, the TCP port can alternatively be configured in a listen point for the PU.

Legacy Configuration

To configure the TCP port in legacy configurations that do not implement a listen point, use the following command in TN3270 server, PU, or DLUR PU configuration modes:

Command	Purpose
`Router(cfg-tn3270)# tcp-port number`	(Optional) Specifies the TCP port (between 0 and 65534) to be used for the PU. The default TCP port number is 23.

Use the **no** form of the command to remove the TCP port from the current configuration scope and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **tcp-port** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Listen-point Configuration

To configure the TCP port in listen-point configurations, use the following command in TN3270 server configuration mode:

Command	Purpose
`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.

Use the **no** form of the command to remove a listen point for the TN3270 server.

Configuring Timing Marks

Configuring the timing marks option for the TN3270 server specifies whether the TN3270 server sends a WILL TIMING-MARK in response to a definite or pacing request by a host application.

The default behavior of the TN3270 server is to send timing marks only for the keepalive function. If you configure the TN3270 server to send timing marks to achieve an end-to-end response protocol, then a WILL TIMING-MARK is sent by the TN3270 server when any of the following conditions are true:

- The host application requests a pacing response.
- The host application requests a definite response (DR), and either the client is not using TN3270E, or the request is not Begin Chain.

The use of timing marks can degrade performance. Some clients do not support timing marks used in this way. Therefore you should only configure timing marks when both of the following conditions are true:

- All clients support this timing mark usage.
- The application benefits from end-to-end acknowledgment.

To configure the timing marks option for the TN3270 server, use the following command in TN3270 server configuration mode:

Command	Purpose
`Router(cfg-tn3270)# timing-mark`	(Optional) Specifies that the TN3270 server sends a WILL TIMING-MARK in response to an application request for a pacing or definite response.

Use the **no** form of the command to disable the sending of WILL TIMING-MARK except as used by the keepalive function.

Configuring the Unbind Action

Configuring the unbind action for the TN3270 server allows you to specify how the TN3270 server responds when it receives an UNBIND request. The TN3270 server can either keep the session or disconnect.

The default behavior in TN3270 server configuration mode is to disconnect the client session upon receipt of an UNBIND. In other configuration modes the default behavior is the currently configured value in the configuration mode applicable to the PU.

To configure the unbind action for the TN3270 server, use the following command in TN3270 server, listen-point, listen-point PU, PU, or DLUR PU configuration modes:

Command	Purpose
`Router(cfg-tn3270)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session disconnects when an UNBIND request is received.

Use the **no** form of the command to remove the unbind action from the current configuration scope and return to the previously configured siftdown value applicable to the PU, or to the default value.

The **unbind-action** command is a siftdown command that is available in multiple command modes. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring SSL Encryption Support

Perform the tasks in the following sections to configure the SSL Encryption feature:

- Obtaining Server Digital Certificate from Certificate Authority, page 40 (Required)
- Loading Server Digital Certificate onto the Flash of the TN3270 Router, page 40 (Required)
- Configuring Security, page 40 (Required)
- Configuring the Profile, page 40 (Required)
- Configuring the Profile Options, page 41 (Optional)

- Configuring the Default Profile, page 41 (Optional)
- Configuring a Listen Point for Security, page 41 (Optional)

Obtaining Server Digital Certificate from Certificate Authority

To obtain a server digital certificate, first create a certificate signing request pointer to the Readme.csr file. The certificate must be in PEM or Base 64 format.

After you obtain the server digital certificate, append the private key file to the digital certificate.

Loading Server Digital Certificate onto the Flash of the TN3270 Router

Copy the digital certificate to the Flash card on the TN3270 router.

Configuring Security

To configure security on the TN3270 server, use the following command beginning in TN3270 server configuration mode:

Command	Purpose
`Router(cfg-tn3270)# security`	Enables security on the TN3270 server and enters security configuration mode.

To enable and disable security on the TN3270 server, use the following commands beginning in security configuration mode:

Command	Purpose
`Router(tn3270-security)# enable`	(Optional) Enables security in the TN3270 server.
`Router(tn3270-security)# disable`	(Optional) Disables the security feature in the TN3270 server.

Configuring the Profile

To configure a security profile on the TN3270 server, use the following command beginning in security configuration mode:

Command	Purpose
`Router(tn3270-security)# profile` *profilename* {**ssl** | **none**}	Specifies a name and a security protocol for a security profile.

Configuring the Profile Options

To configure the security profile options, use the following commands beginning in profile configuration mode:

Command	Purpose
`Router(tn3270-sec-profile)# keylen {40 | 128}`	Specifies the maximum bit length for the session encryption key for the TN3270 server.
`Router(tn3270-sec-profile)# encryptorder [DES] [3DES] [RC4] [RC2] [RC5]`	Specifies the encryption algorithm for the TN3270 SSL Encryption Support.
`Router(tn3270-sec-profile)# servercert` *location*	Specifies the location of the TN3270 server's security certificate in the Flash memory. This command reads the security certificate from the specified location.
`Router(tn3270-sec-profile)# certificate reload`	(Optional) Reads the profile security certificate from the file specified in the **servercert** command.

Configuring the Default Profile

To configure the default security profile name to be applied to the listen-points, use the following command beginning in security configuration mode:

Note The **profile** command must be specified before configuring a default-profile.

Command	Purpose
`Router(tn3270-security)# default-profile` *profilename*	Specifies the name of the profile to be applied to the listen-points by default.

Configuring a Listen Point for Security

To configure a listen-point for security, use the following command beginning in TN3270 listen-point configuration mode:

Note The **sec-profile** command is optional if the **default-profile** command has been configured.

Command	Purpose
`Router(tn3270-lpoint)# sec-profile` *profilename*	Specifies the security profile to be associated with a listen-point.

Configuring the TN3270 Server with LU Pooling

This section describes the required tasks to configure the TN3270 server with LU pooling in an APPN environment using DLUR PUs and in a non-APPN environment using direct PUs.

Step 1 Before configuring the TN3270 server, follow the "Guidelines for Configuring LU Pooling" section on page 43.

Step 2 Before you begin configuring the TN3270 server, be sure that you have configured host connectivity to the router. For more information about configuring host connectivity, see the "Configuring Host Connections" section on page 19.

Step 3 Complete the following tasks to configure the TN3270 server with LU pooling in an APPN environment using DLUR:

- Configuring the TN3270 Server and Defining a Pool, page 43
- Configuring DLUR, page 44
- Configuring SAPs Under DLUR, page 45
- Configuring a Listen Point and Nailing Clients to Pools, page 45
- Configuring Inverse DNS Nailing, page 46
- Configuring a Listen-Point PU to Define DLUR PUs and Allocate LUs, page 48
- Configuring a Listen-Point PU to Define DLUR PUs using Dynamic LU Naming, page 49

Note You can also use DLUR to reach a mix of APPN and non-APPN hosts. The host owning the PUs must be an APPN network node that also supports the subarea (that is, an interchange node). When an SLU starts a session with any of the APPN hosts, it can use session switching to reach that host directly. When it starts a session with a non-APPN host, the traffic will be routed through the owning host.

Step 4 Complete the following tasks to configure the TN3270 server with LU pooling in a non-APPN environment:

- Configuring the TN3270 Server and Defining a Pool, page 50
- Configuring a Listen Point and Nailing Clients to Pools, page 51
- Configuring a Listen-Point PU to Define Direct PUs and Allocate LUs, page 52
- Configuring a Listen-Point PU to Define Direct PUs using Dynamic LU Naming, page 53

Note The differences between the configuration tasks in a non-APPN environment and the APPN configuration tasks are that you do not configure DLUR or SAPs under DLUR, and you configure direct PUs at the listen point instead of DLUR PUs. All other options are the same.

Refer to the "Configuring the TN3270 Server Options" section on page 30 of this publication and the "TN3270 Server Commands" chapter of the *Cisco IOS Bridging and IBM Networking Command Reference* (Volume 2 of 2) for additional information about the commands described in this section and chapter.

Guidelines for Configuring LU Pooling

To configure LU pools on the TN3270 server on a CMCC adapter, perform the following tasks:

1. Define a pool using the **pool** command.
2. Allocate specific LOCADDRs or LUs to the pool using the **allocate lu** command.
3. (Optional) Nail clients to the pool using the **client ip pool** command.

When configured, the pool becomes one of the several criteria used by the TN3270 server to assign an LU to a client. When a client requests a connection, the TN3270 server determines the authorized capabilities of the client. For example, the TN3270 server attempts to determine whether LU nailing definitions exist for the client.

Client preferences are taken into consideration. Examples of client preferences are:

- Device name on CONNECT request (TN3270E)
- LU name on **TERMINAL-TYPE** command (RFC 1576)
- Model type

When the client criteria is processed, the TN3270 server assigns the first available LU in the group to the client. If an appropriate LU is not found, the TN3270 connection is closed.

For more information about LU allocation in the TN3270 server, see the "LU Allocation" section on page 7. For an example of how LUs are allocated within LU pools, see the "LU Pooling Configuration Example" section on page 66.

Configuring the TN3270 Server and Defining a Pool

To establish a TN3270 server on the internal LAN interface on the CMCC adapter and configure LU pooling, use the following commands beginning in global configuration mode. When you use the **tn3270-server** command, you enter TN3270 server configuration mode and can use all other commands in the task list.

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the TN3270 server and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*Port* value corresponds to the virtual interface, which is port 2. • CPA—*Port* value corresponds to port 0.
Step 2	`Router(config-if)# tn3270-server`	Specifies a TN3270 server on the internal LAN interface and enters TN3270 server configuration mode.
Step 3	`Router(cfg-tn3270)# pool poolname [cluster layout [layout-spec-string]]`	Defines clusters of LUs and allocates LOCADDRs.
Step 4	`Router(cfg-tn3270)# generic-pool {permit | deny}`	(Optional) Selects whether "leftover" LUs can be used from a generic LU pool.
Step 5	`Router(cfg-tn3270)# idle-time seconds`	(Optional) Specifies the idle time for server disconnect.

	Command	Purpose
Step 6	`Router(cfg-tn3270)# ip precedence {screen | printer} value`	(Optional) Specifies the precedence level for IP traffic in the TN3270 server.
Step 7	`Router(cfg-tn3270)# ip tos {screen | printer} value`	(Optional) Specifies the ToS level for IP traffic in the TN3270 server.
Step 8	`Router(cfg-tn3270)# keepalive seconds [send {nop | timing-mark [max-response-time]}]`	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 9	`Router(cfg-tn3270)# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 10	`Router(cfg-tn3270)# lu termination {termself | unbind}`	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off or disconnects a device.
Step 11	`Router(cfg-tn3270)# maximum-lus number`	(Optional) Specifies the maximum number (between 0 and 32000) of LU control blocks allowed for the TN3270 server. The default is 2100.
Step 12	`Router(cfg-tn3270)# client [ip [ip-mask]] lu maximum number`	(Optional) Specifies the maximum number (between 0 and 65535) of LU sessions allowed for a client at an IP address or IP subnet address.
Step 13	`Router(cfg-tn3270)# timing-mark`	(Optional) Specifies that the TN3270 server sends a WILL TIMING-MARK in response to an application request for a pacing or definite response.
Step 14	`Router(cfg-tn3270)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.

Configuring DLUR

This task is required when configuring DLUR connected hosts. To configure DLUR parameters for the TN3270 server, use the following commands beginning in TN3270 server configuration mode:

	Command	Purpose
Step 1	`Router(cfg-tn3270)# dlur fq-cpname fq-dlusname`	Creates a DLUR function in the TN3270 server and enters DLUR configuration mode.
Step 2	`Router(tn3270-dlur)# dlus-backup dlusname2`	(Optional) Specifies a backup DLUS for the DLUR function.
Step 3	`Router(tn3270-dlur)# preferred-nnserver NNserver`	(Optional) Specifies the preferred network node (NN) server.

Configuring SAPs Under DLUR

To configure SAPs under the DLUR function, use the following commands beginning in DLUR configuration mode:

	Command	Purpose
Step 1	`Router(tn3270-dlur)# lsap type adapno [lsap]`	Creates a SAP function under DLUR and enters DLUR SAP configuration mode.
Step 2	`Router(tn3270-dlur-lsap)# vrn vrn-name`	(Optional) Identifies an APPN virtual routing node (VRN).
Step 3	`Router(tn3270-dlur-lsap)# link name [rmac rmac] [rsap rsap]`	(Optional) Creates named links to hosts. A link should be configured to each potential NN server. (The alternative is to configure the NN servers to connect to DLUR.) If VRN is used it is not necessary to configure links to other hosts. Do not configure multiple links to the same host.

Configuring a Listen Point and Nailing Clients to Pools

To configure a listen point on the internal LAN interface on the CMCC adapter and nail clients to pools, use the following commands beginning in TN3270 server configuration mode.

When you use the **listen-point** command, you enter listen-point configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands in listen-point configuration mode will override values that you previously entered in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# client ip ip-address [mask] pool poolname`	Nails a client located at the IP address or subnet to a pool.
Step 3	`Router(tn3270-lpoint)# generic-pool {permit | deny}`	(Optional) Selects whether “leftover” LUs can be used from a generic LU pool.
Step 4	`Router(tn3270-lpoint)# idle-time seconds`	(Optional) Specifies the idle time for server disconnect.
Step 5	`Router(tn3270-lpoint)# ip precedence {screen | printer} value`	(Optional) Specifies the precedence level for IP traffic in the TN3270 server.
Step 6	`Router(tn3270-lpoint)# ip tos {screen | printer} value`	(Optional) Specifies the ToS level for IP traffic in the TN3270 server.

	Command	Purpose
Step 7	`Router(tn3270-lpoint)# keepalive` *seconds* `[send {nop | timing-mark [`*max-response-time*`]}]`	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 8	`Router(tn3270-lpoint)# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 9	`Router(tn3270-lpoint)# lu termination {termself | unbind}`	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off or disconnects a device.
Step 10	`Router(tn3270-lpoint)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.

Configuring Inverse DNS Nailing

Perform the tasks in the following section to configure the different methods of Inverse DNS Nailing feature:

- Nailing Clients to Pools by IP Address, page 46
- Nailing Clients to Pools by Device Name, page 47
- Nailing Clients to Pools by Device Name using a Domain ID, page 47
- Nailing Clients to Pools by Domain Name, page 47
- Nailing Clients to Pools by Domain Name Using a Domain ID, page 48

Note You can configure Inverse DNS Nailing five different ways by using the same commands. This task table section presents the five different configuration methods as separate task tables.

Use the **domain-id** command only when you are going to configure the **client pool** command with the **name** keyword and *DNS-domain-identifier* option specified or with the **domain-id** keyword specified.

Nailing Clients to Pools by IP Address

To nail a client to a pool of LUs by IP address, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# client ip ip-address [ip-mask] pool poolname`	Nails a client located at the IP address to a pool.

Nailing Clients to Pools by Device Name

To nail a client to a pool of LUs by device name, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# client name DNS-name pool poolname`	Nails a client located at the DNS device name to a pool.

Nailing Clients to Pools by Device Name using a Domain ID

To nail a client to a pool of LUs by device name using a domain ID, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# domain-id DNS-domain-identifier DNS-domain`	(Optional) Specifies a domain name suffix to be appended to the configured machine names to form a fully qualified name.
Step 2	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 3	`Router(tn3270-lpoint)# client name DNS-name DNS-domain-identifier pool poolname`	Nails a client located at the IP address to a pool.

Nailing Clients to Pools by Domain Name

To nail a client to a pool of LUs by domain name, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# client domain-name DNS-domain pool poolname`	Nails a client located at the domain-name to a pool.

Nailing Clients to Pools by Domain Name Using a Domain ID

To nail a client to a pool of LUs by domain name using a domain ID, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# domain-id DNS-domain-identifier DNS-domain`	(Optional) Specifies a domain name suffix to be appended to the configured machine names to form a fully qualified name.
Step 2	`Router(cfg-tn3270)# listen-point ip-address [tcp-port [number]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 3	`Router(tn3270-lpoint)# client domain-id DNS-domain-identifier pool poolname`	Nails a client located at the domain ID to a pool.

Configuring a Listen-Point PU to Define DLUR PUs and Allocate LUs

To configure a listen-point PU on the internal LAN interface on the CMCC adapter and define DLUR PUs, use the following commands beginning in listen-point configuration mode.

When you use the **pu** command, you enter listen-point PU configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands in listen-point PU configuration mode will override values that you previously entered in listen-point or TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(tn3270-lpoint)# pu pu-name idblk-idnum dlur`	Creates a DLUR PU. This command changes the configuration mode from listen-point to listen-point PU.
Step 2	`Router(tn3270-lpoint-pu)# allocate lu lu-address pool poolname clusters count`	Assigns LUs to the pool beginning with the LOCADDR specified by *lu-address* for a total of *count* LUs.
Step 3	`Router(tn3270-lpoint-pu)# generic-pool {permit | deny}`	(Optional) Selects whether “leftover” LUs can be used from a generic LU pool.
Step 4	`Router(tn3270-lpoint-pu)# idle-time seconds`	(Optional) Specifies the idle time for server disconnect.

	Command	Purpose
Step 5	`Router(tn3270-lpoint-pu)# keepalive` *seconds* `[send {nop | timing-mark [`*max-response-time*`]}]`	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 6	`Router(tn3270-lpoint-pu)# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 7	`Router(tn3270-lpoint-pu)# lu termination {termself | unbind}`	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off or disconnects a device.
Step 8	`Router(tn3270-lpoint-pu)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.

Configuring a Listen-Point PU to Define DLUR PUs using Dynamic LU Naming

To configure a listen-point PU on the internal LAN interface on the CMCC adapter, and to define DLUR PUs using dynamic LU naming, use the following commands beginning in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point` *ip-address* `[tcp-port [`*number*`]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# pu` *pu-name idblk-idnum* `dlur [lu-seed` *lu-name-stem*`]`	Creates a DLUR PU and enters listen-point PU configuration mode. The **lu-seed** optional keyword specifies the LU name that the client uses when a specific LU name request is needed.
Step 3	`Router(tn3270-lpoint-pu)# lu deletion {always | normal | non-generic | never | named}`	Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects. **Note** You must specify the **named** option when configuring dynamic LU naming on the PU.

When you use the **pu** command, you enter listen-point PU configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands (such as the **lu deletion** command) in listen-point PU configuration mode will override values that you previously entered in listen-point or TN3270 server configuration mode. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Configuring the TN3270 Server and Defining a Pool

To establish a TN3270 server on the internal LAN interface on the CMCC adapter and configure LU pooling, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface channel slot/port`	Selects the interface on which to configure the TN3270 server and enters interface configuration mode. The *port* value differs by the type of CMCC adapter: • CIP—*port* value corresponds to the virtual interface, which is port 2. • CPA—*port* value corresponds to port 0.
Step 2	`Router(config-if)# tn3270-server`	Specifies a TN3270 server on the internal LAN interface and enters TN3270 server configuration mode.
Step 3	`Router(cfg-tn3270)# pool poolname [cluster layout [layout-spec-string]]`	Defines clusters of LUs and allocates LOCADDRs.
Step 4	`Router(cfg-tn3270)# idle-time seconds`	(Optional) Specifies the idle time for server disconnect.
Step 5	`Router(cfg-tn3270)# keepalive seconds [send {nop | timing-mark [max-response-time]}]`	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 6	`Router(cfg-tn3270)# ip precedence {screen | printer} value`	(Optional) Specifies the precedence level for IP traffic in the TN3270 server.
Step 7	`Router(cfg-tn3270)# ip tos {screen | printer} value`	(Optional) Specifies the ToS level for IP traffic in the TN3270 server.
Step 8	`Router(cfg-tn3270)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.
Step 9	`Router(cfg-tn3270)# generic-pool {permit | deny}`	(Optional) Selects whether “leftover” LUs can be used from a generic LU pool.
Step 10	`Router(cfg-tn3270)# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 11	`Router(cfg-tn3270)# lu termination {termself | unbind}`	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off or disconnects a device.

Configuring a Listen Point and Nailing Clients to Pools

To configure a listen point on the internal LAN interface on the CMCC adapter and nail clients to pools, use the following commands beginning in TN3270 server configuration mode.

When you use the **listen-point** command, you enter listen-point configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands in listen-point configuration mode will override values that you previously entered in TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point` *ip-address* [**tcp-port** [*number*]]	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# client ip` ip-address [*mask*] **pool** *poolname*	Nails a client located at the IP address or subnet to a pool.
Step 3	`Router(tn3270-lpoint)# idle-time` *seconds*	(Optional) Specifies the idle time for server disconnect.
Step 4	`Router(tn3270-lpoint)# keepalive` *seconds* [**send** {**nop** | **timing-mark** [*max-response-time*]}]	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 5	`Router(tn3270-lpoint)# ip precedence` {**screen** | **printer**} *value*	(Optional) Specifies the precedence level for IP traffic in the TN3270 server.
Step 6	`Router(tn3270-lpoint)# ip tos` {**screen** | **printer**} *value*	(Optional) Specifies the ToS level for IP traffic in the TN3270 server.
Step 7	`Router(tn3270-lpoint)# unbind-action` {**keep** | **disconnect**}	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.
Step 8	`Router(tn3270-lpoint)# generic-pool` {**permit** | **deny**}	(Optional) Selects whether “leftover” LUs can be used from a generic LU pool.
Step 9	`Router(tn3270-lpoint)# lu deletion` {**always** | **normal** | **non-generic** | **never**}	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 10	`Router(tn3270-lpoint)# lu termination` {**termself** | **unbind**}	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off or disconnects a device.

Configuring a Listen-Point PU to Define Direct PUs and Allocate LUs

To configure a listen-point PU on the internal LAN interface on the CMCC adapter and configure direct PUs, use the following commands beginning in listen-point configuration mode.

When you use the **pu** command, you enter listen-point PU configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands in listen-point PU configuration mode will override values that you previously entered in listen-point or TN3270 server configuration mode.

	Command	Purpose
Step 1	`Router(tn3270-lpoint)# pu pu-name idblk-idnum type adapter-number lsap [rmac rmac] [rsap rsap] [lu-seed lu-name-stem]`	Creates a direct PU. This command changes the configuration mode from listen-point to listen-point PU.
Step 2	`Router(tn3270-lpoint-pu)# allocate lu lu-address pool poolname clusters count`	Assigns LUs to the pool beginning with the LOCADDR specified by *lu-address* for a total of *count* LUs.
Step 3	`Router(tn3270-lpoint-pu)# idle-time seconds`	(Optional) Specifies the idle time for server disconnect.
Step 4	`Router(tn3270-lpoint-pu)# keepalive seconds [send {nop | timing-mark [max-response-time]}]`	(Optional) Specifies the following keepalive parameters: • Number of seconds of inactivity to elapse before the TN3270 server transmits a DO TIMING-MARK or Telnet nop to the TN3270 client. • Maximum time within which the TN3270 server expects a response to the DO TIMING-MARK from the TN3270 client before the server disconnects.
Step 5	`Router(tn3270-lpoint-pu)# unbind-action {keep | disconnect}`	(Optional) Specifies whether the TN3270 session will disconnect when an UNBIND request is received.
Step 6	`Router(tn3270-lpoint-pu)# generic-pool {permit | deny}`	(Optional) Selects whether “leftover” LUs can be used from a generic LU pool.
Step 7	`Router(tn3270-lpoint-pu)# lu deletion {always | normal | non-generic | never}`	(Optional) Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects.
Step 8	`Router(tn3270-lpoint-pu)# lu termination {termself | unbind}`	(Optional) Specifies the type of termination request that is sent by the TN3270 server when a client turns off his device or disconnects.

Configuring a Listen-Point PU to Define Direct PUs using Dynamic LU Naming

To configure a listen-point PU on the internal LAN interface on the CMCC adapter and configure direct PUs using dynamic LU naming, use the following commands beginning in listen-point configuration mode.

	Command	Purpose
Step 1	`Router(cfg-tn3270)# listen-point` *ip-address* `[tcp-port` `[`*number*`]]`	Specifies the IP address and TCP port number to create a listen point. The default TCP port number is 23. This command changes the configuration mode from TN3270 to listen-point.
Step 2	`Router(tn3270-lpoint)# pu` *pu-name idblk-idnum type adapter-number lsap* `[rmac` *rmac*`] [rsap` *rsap*`] [lu-seed` *lu-name-stem*`]`	Creates a direct PU and enters listen-point PU configuration mode. The **lu-seed** optional keyword specifies the LU name that the client uses when a specific LU name request is needed.
Step 3	`Router(tn3270-lpoint-pu)# lu deletion {always | normal | non-generic | never | named}`	Specifies whether the TN3270 server sends a REPLY-PSID poweroff request to VTAM to delete the corresponding LU when a client disconnects. **Note** You must specify the **named** option when configuring dynamic LU naming on the PU.

When you use the **pu** command, you enter listen-point PU configuration mode and can use all other commands in this task list. Values that you enter for siftdown commands (such as the **lu deletion** command) in listen-point PU configuration mode will override values that you previously entered in listen-point or TN3270 server configuration mode. For more information about configuring siftdown commands, see the "Configuring TN3270 Siftdown Commands" section on page 28.

Migrating from Legacy TN3270 Server Configuration Methods

Prior to Cisco IOS Release 12.0(5)T, TN3270 server configuration did not directly support listen points and LU pool configurations. These earlier methods for configuring PUs are referred to as "legacy" configuration methods. The TN3270 server commands to configure PUs vary slightly depending on whether or not you are using legacy configuration methods or listen points and LU pooling to configure PUs. While the legacy TN3270 server configuration commands are still supported, it is important to understand these variations in configuration so that you are not confused by the similar, but distinct command usages implemented for LU pooling.

Note Be sure that you use only a single configuration method for any particular IP address. Do not configure the same IP address using legacy methods and the newer listen-point configuration methods.

Methods of Configuring Direct PUs

For example, there are two ways in which you can configure direct PUs in the TN3270 server:

- TN3270 server configuration—In this legacy configuration mode you can use the **pu** (TN3270) command with the *ip-address* argument to create a PU entity that has its own direct link to a host at that IP address.
- Listen-point configuration—In this configuration mode you can use a different version of the **pu** command, but without an *ip-address* argument, to also create a PU entity that has its own direct link to a host defined at the listen point. In this configuration scenario, the IP address of the host is defined using the **listen-point** command and not in the **pu** (listen-point) command. This usage of direct PU configuration at a listen point allows you to eliminate repetitive configuration of the host IP address for each PU.

For examples of these methods of direct PU configuration see the "Basic Configuration Example" section on page 64 and the "Listen-Point Direct PU Configuration Example" section on page 65.

Methods of Configuring DLUR PUs

Similarly, there are also two ways in which you can configure DLUR PUs in the TN3270 server:

- DLUR configuration—In this legacy configuration mode you can use a version of the **pu** command—**pu** (DLUR)—with *pu-name*, *idblk-idnum*, and *ip-address* arguments to create a PU entity that uses the SNA session switching facility to communicate with a host.
- Listen-point configuration—In this configuration mode you use a different command—the **pu dlur** command—with *pu-name* and *idblk-idnum* arguments to create a PU entity that uses the SNA session switching facility to communicate with a host addressed at the listen point.

For an example of these methods of DLUR PU configuration see the "Listen-Point DLUR PU Configuration Example" section on page 65.

Methods of LU Nailing

LU nailing is a method by which you can associate a client's connection request with a specific LU or pool of LUs. Use the following different methods to nail LUs in the TN3270 server:

- Nailing Clients to Specific LUs, page 35
- Nailing Clients to Pools, page 35
- Using a Combination of Nailing Methods, page 55

Nailing Clients to Specific LUs

Use the **client ip lu** legacy command when you want to assign a specific LOCADDR to a particular client at an IP address or subnet. This method of nailing is useful when a particular client must use a specific LU. You can use the **client printer ip lu** command to assign a particular LOCADDR to a client printer at an IP address or subnet.

Nailing Clients to Pools

Use the **client ip pool** command in listen-point configuration mode when you want to assign a group of LUs from a pool defined in the TN3270 server for a client at an IP address or subnet. This method of nailing is useful when a client needs to have one of a group LUs associated with a particular PU.

This configuration method uses the **allocate lu** listen-point PU configuration command to assign the range of LOCADDRS to the pool. The **pool** command defines the pool as a cluster of screen and printer LUs. In this method, clients can use the ASSOCIATE request to access printers defined to the pool.

Using a Combination of Nailing Methods

You can use both methods of LU nailing in a particular TN3270 server configuration, but there is no precedence in the configuration statements. Therefore when you nail a client to a specific LU or to a pool, you must be sure that the LOCADDR has not already been allocated. You cannot specify the same LOCADDR in both an individual LU nailing statement and in a pool. The CMCC adapter does not allow a LOCADDR to be allocated multiple times, so the LU allocations in the TN3270 server must not overlap.

For example, the following configuration statements are in error because LU 5 is allocated to both the pool and to an individual client at IP address 10.20.30.40:

```
tn3270-server
 pool MYPOOL cluster layout 4s1p
 pu PU1 12345678 tok 0 10
 allocate lu 5 pool MYPOOL clusters 2
 client ip 10.20.30.40 lu 5
```

The following example shows a valid configuration where a client at IP address 10.20.30.40 is nailed to the pool named EXAMPLE, which is allocated LOCADDRs 1 through 10, and an individual client at IP address 10.20.30.50 that is nailed only to LU 150:

```
tn3270-server
 pool EXAMPLE cluster layout 2s2p
 listen-point 80.80.80.81
  client ip 10.20.30.40 pool EXAMPLE
  pu PU1 12345678 tok 0 10
   allocate lu 1 pool EXAMPLE clusters 10
   client ip 10.20.30.50 lu 150
```

Verifying the TN3270 Server Configuration

This section provides basic steps that you can use to verify TN3270 server configurations. For detailed examples of configuration verification procedures for specific TN3270 server scenarios, see the Cisco *TN3270 Design and Implementation Guide*.

- Verify a Server Configuration that Uses LU Pooling, page 55
- Verify Dynamic LU Naming on the TN3270 Server, page 56
- Verifying Inverse DNS Nailing on the TN3270 Server, page 58
- Verifying SSL Encryption Support on the TN3270 Server, page 59

Verify a Server Configuration that Uses LU Pooling

Step 1 To display the current router configuration, enter the **show run** command:

```
router#show run
Building configuration...

interface Channel6/1
 no ip address
 no keepalive
 csna E160 40
!
interface Channel6/2
 ip address 172.18.4.17 255.255.255.248
 no keepalive
 lan TokenRing 15
```

```
 source-bridge 15 1 500
 adapter 15 4000.b0ca.0015
lan TokenRing 16
 source-bridge 16 1 500
 adapter 16 4000.b0ca.0016
tn3270-server
 pool PCPOOL   cluster layout 4s1p
 pool SIMPLE   cluster layout 1a
 pool UNIXPOOL cluster layout 49s1p
 dlur NETA.SHEK NETA.MVSD
  lsap token-adapter 15 04
   link SHE1     rmac 4000.b0ca.0016
 listen-point 172.18.4.18 tcp-port 23
  pu PU1      91903315 dlur
   allocate lu 1 pool PCPOOL   clusters 10
   allocate lu 51 pool UNIXPOOL clusters 2
   allocate lu 200 pool SIMPLE   clusters 50
 listen-point 172.18.4.19 tcp-port 2023
  pu PU2      91913315 token-adapter 16 08
   allocate lu 1 pool UNIXPOOL clusters 2
   allocate lu 101 pool SIMPLE   clusters 100
   allocate lu 201 pool PCPOOL   clusters 10
```

Step 2 To display information about the client LUs associated with a specific PU including the cluster layout and pool name, enter the **show extended channel tn3270-server pu** command:

```
Router#show extended channel 6/2 tn3270-server pu pu1 cluster

name(index)    ip:tcp               xid   state     link   destination  r-lsap
PU1(1)       172.18.4.18:23      91903315 ACTIVE    dlur   NETA.SHPU1
idle-time    0      keepalive 1800      unbind-act discon   generic-pool  perm
ip-preced-screen 0 ip-preced-printer 0 ip-tos-screen  0 ip-tos-printer 0
lu-termination unbind lu-deletion never
bytes 27489 in, 74761 out; frames 1164 in, 884 out; NegRsp 0 in, 0 out
actlus 5, dactlus 0, binds 5
Note: if state is ACT/NA then the client is disconnected

lu    name   client-ip:tcp       nail state    cluster   pool  count
1   SHED1001 161.44.100.162:1538   N   ACT/SESS 1/4s1p  PCPOOL   1/5
51  SHED1051 161.44.100.162:1539   N   ACT/SESS 1/49s1p UNIXPOOL 1/50
151 SHED1151 161.44.100.162:1536   N   ACT/SESS 1/1a    :GENERIC 1/1
152 SHED1152 161.44.100.162:1537   N   ACT/SESS 1/1a    :GENERIC 1/1
200 SHED1200 161.44.100.162:1557   N   ACT/SESS 1/1a    SIMPLE   1/1
```

Verify Dynamic LU Naming on the TN3270 Server

Complete the following steps to verify the Dynamic LU Naming enhancement:

Step 1 Issue the **show extended channel tn3270-server** command. Confirm that lu-deletion is set to **named**.

```
Router# show extended channel 3/2 tn3270-server

<current stats> < connection stats >  <response time(ms)>
server-ip:tcp        lu in-use   connect disconn fail   host     tcp
172.28.1.106:23     510     1       12       11     0     54     40
172.28.1.107:23     511     0        0        0     0      0      0
172.28.1.108:23     255     0        0        0     0      0      0
total              1276     1
configured max_lu 20000
```

```
idle-time    0           keepalive 1800      unbind-action disconnect
tcp-port   23            generic-pool permit no timing-mark
lu-termination unbind lu-deletion named
```

Step 2 To verify that dynamic LU naming is configured on the PU named **PU1**, issue the **show extended channel tn3270-server pu** command. Confirm that lu-deletion is set to **named**.

```
Router# show extended channel 6/2 tn3270-server pu pu1

name(index)    ip:tcp               xid   state     link   destination r-lsap
PU1(1)       172.18.4.18:23      91903315 ACTIVE    dlur   NETA.SHPU1

idle-time    0      keepalive 1800      unbind-act discon   generic-poolperm
ip-preced-screen 0 ip-preced-printer 0 ip-tos-screen  0 ip-tos-printer 0
lu-termination unbind lu-deletion named
```

Troubleshooting Tips for Dynamic LU Naming

To troubleshoot dynamic LU naming, use the following tips:

- You must replace the default exit ISTEXCSD with the VTAM User Exit for TN3270 Name Pushing, which you can download from the IBM website: http://www.ibm.com. This exit causes VTAM to ignore the LUSEED parameter on the PU statement, and instead use the SLU name sent by the router in the subvector 86 when a client connects in. If you do not configure this exit, VTAM ignores the subvector 86 and the specified LU name.
- If the LUSEED operand is specified on the mainframe, but the subvector 86 requires an LU name, the VTAM User Exit for TN3270 Name Pushing ignores the LUSEED operand.
- If the LUSEED operand is not specified on the mainframe, and the subvector 86 is not present, then the VTAM User Exit for TN3270 Name Pushing cannot generate an LU name. VTAM does not log this failure, and the TN3270 server does not receive the ACTLU request. The TN3270 server displays the following message:

```
*Apr 17 12:40:53:%CIP2-3-MSG:slot2 :
%TN3270S-3-NO_DYN_ACTLU_REQ_RCVD
  No ACTLU REQ received on LU JJDL1.6
```

Specify the INCLUD0E=YES parameter on VTAM so that the TN3270 server will always receive the LU name generated by the VTAM User Exit for TN3270 Name Pushing.

Verifying Inverse DNS Nailing on the TN3270 Server

Complete the following steps to verify the Inverse DNS Nailing enhancement:

Step 1 To list all nailing statements with a specific nailed-domain name, enter the **show extended channel tn3270-server nailed-domain** command:

```
Router# show extended channel 1/2 tn3270-server nailed-domain .cisco.com
CISCO.COM listen-point 172.18.4.18  pool PCPOOL
```

Step 2 To list all nailing statements with a specific nailed machine name, enter the **show extended channel tn3270-server nailed-name** command:

```
Router# show extended channel 1/2 tn3270-server nailed-name myclient.cisco.com
MYCLIENT.CISCO.COM     listen-point 172.18.4.18  pool PCPOOL
HISCLIENT.CISCO.COM    listen-point 172.18.4.18  pool UNIXPOOL
HERCLIENT.CISCO.COM    listen-point 172.18.4.19  pool GENERALPOOL
```

Troubleshooting Tips for Inverse DNS Nailing

To troubleshoot inverse DNS nailing, use the following tips:

- If an inverse DNS lookup fails it could be because the DNS server is unavailable (either because it was not configured, or because it is down). In this case, you cannot tell if the client is nailed because it does not have a name. To complicate the scenario, assume there was not a legacy nailing match, but the PU supports LUs that have been assigned from a generic pool. In this situation, the client disconnects and the router displays the following console message:

  ```
  A connection attempt from client <ip address> was refused because its DNS name could
  not be obtained.
  ```

  This action removes any potential security risk but presents potential disadvantages—the client could be denied a valid LU, and the generic-pool permit and deny settings might be ignored. For these reasons, it is strongly recommended that users configure the Inverse DNS Nailing enhancement on a PU that does *not* support LUs that have been assigned from a generic pool or a PU that has the **generic-pool** command configured with the **deny** keyword specified.
- If an inverse DNS lookup succeeds, but the name is not nailed or the client has no machine name, then the client is not nailed and the TN3270 server reverts to the legacy LU nailing process.

Verifying SSL Encryption Support on the TN3270 Server

Complete the following steps to verify the SSL Encryption Support enhancement:

Step 1 To verify the security profile on the TN3270 server, enter the **show extended channel tn3270-server security** command using the **sec-profile** option. Confirm that the status is enabled (status: ENABLE), and that the security certificate is loaded (Certificate Loaded: YES).

```
Router# show extended channel 3/2 tn3270-server security sec-profile cert40
status:ENABLE Default Profile: (Not Configured)
Name               Active LUs  keylen encryptorder            Mechanism
CERT40                     0     40    RC4 RC2 RC5 DES 3DES    SSL
Servercert:slot0:coach188.pem
Certificate Loaded:YES Default-Profile:NO
```

Step 2 To verify the security profile on the TN3270 server listen-point, enter the **show extended channel tn3270-server security** command using the **listen-point** option. Confirm that the status is enabled (status: ENABLE) and that the state is active (State ACTIVE).

```
Router# show extended channel 3/2 tn3270-server security listen-point 172.18.5.188
status:ENABLE Default Profile: (Not Configured)
IPaddress      tcp-port   Security-Profile   active-sessions  Type    State
172.18.5.188    23        CERT40               0              Secure  ACTIVE
Active Sessions using Deleted Profile:0
```

Configuring the TN3270 Server for Response-Time Monitoring

To configure client subnet response-time groups, use the following commands in response-time configuration mode:

	Command	Purpose
Step 1	`Router(tn3270-resp-time)# response-time group` *name* [`bucket boundaries` *t1 t2 t3 t4*] [`multiplier` *m*]	Configures the client subnet response-time group.
Step 2	`Router(tn3270-resp-time)# client ip` *ip-address* [*ip-mask*]	Specifies the IP address of the subnet being added to this client group.

Verifying Response-Time Configuration

To verify the configuration of the client subnet response-time groups, use the **show extended channel tn3270-server response-time subnet** command.

To display a complete list of client subnet groups and their response-time collection control parameters, use the following form of the command:

```
Router# show extended channel 3/2 tn3270-server response-time subnet
group SUBNETGROUP1
  subnet 10.10.10.0 255.255.255.192
  aggregate NO excludeip NO dynamic definite response NO
  sample period multiplier 30
  bucket boundaries 10 20 50 100
group SUBNETGROUP2
  subnet 10.10.10.128 255.255.255.192
  subnet 10.10.10.192 255.255.255.192
  aggregate NO exclude ip NO dynamic definite response NO
  sample period multiplier 40
  bucket boundaries 20 30 60 120
group CLIENT SUBNET OTHER
  aggregate NO exclude ip NO dynamic definite response NO
  sample period multiplier 30
  bucket boundaries 10 20 50 100
```

To display the response-time collection parameters for a specific subnet, along with a list of the client members and their response-time statistics, use the following form of the command:

```
Router# show extended channel 3/2 tn3270-server response-time subnet
10.10.10.0 255.255.255.192 detail

group SUBNETGROUP1
  subnet 10.10.10.0 255.255.255.192
  aggregate NO excludeip NO dynamic definite response NO
  sample period multiplier 30
  bucket boundaries 10 20 50 100
  client 10.10.10.129:23
    buckets 5 8 11 9 4
    average total response time 33 average IP response time 24
    number of transactions 37
  client 10.10.10.130:23
    buckets 6 9 10 10 2
    average total response time 32 average IP response time 25
    number of transactions 37
  client 10.10.10.131:23
    buckets 11 14 10 8 7
    average total response time 27 average IP response time 19
    number of transactions 50
```

Monitoring and Maintaining the TN3270 Server

Use the following **show** commands in the privileged EXEC mode to monitor the TN3270 server. The *port* value differs by the type of CMCC adapter:

- CIP—*port* value corresponds to the virtual interface, which is port 2
- CPA—*port* value corresponds to port 0

Command	Purpose
Router# **show extended channel** *slot/port* **tn3270-server**	Displays the current server configuration parameters and the status of the PUs defined in each server.
Router# **show extended channel** *slot/port* **tn3270-server client-ip-address** *ip-address* [**disconnected** | **in-session** | **pending**]	Displays information about all clients at a specific IP address.
Router# **show extended channel** *slot/port* **tn3270-server dlur**	Displays information about the SNA session switch.
Router# **show extended channel** *slot/port* **tn3270-server dlurlink** *name*	Displays information about the DLUR components.
Router# **show extended channel** *slot/port* **tn3270-server nailed-ip** *ip-address*	Displays mappings between a nailed client IP address and nailed LUs.
Router# **show extended channel** *slot/virtual channel* **tn3270-server pu** *pu-name* [**cluster**]	Displays information about the client LUs associated with a specified PU including the cluster layout and pool name.
Router# **show extended channel tn3270-server pu** *pu-name* **lu** *lu-number* [**history**]	Displays the status of the LU.
Router# **show extended channel** *slot/port* **tn3270-server response-time application** [*appl-name* [**detail**]]	Displays information about each client group application for the specified VTAM appl name. List each member of the client group with its individual response-time statistics.
Router# **show extended channel** *slot/port* **tn3270-server response-time global**	Displays information about the global client groups.
Router# **show extended channel** slot/port **tn3270-server response-time link** [*link-name*]	Displays information about the specified per-host-link client group.
Router# **show extended channel** slot/port **tn3270-server response-time listen-point**	Displays information about listen-point type client groups.
Router# **show extended channel** slot/port **tn3270-server response-time subnet** [*ip-address ip-mask* [**detail**]]	Displays information about the specified client group.

Other maintenance and monitoring options for the TN3270 include:

- Managing DLUR Links, page 62
- Monitoring Dynamic LU Naming, page 63
- Monitoring Inverse DNS Nailing, page 63
- Shutting Down the TN3270 Server and Its Entities, page 63

Managing DLUR Links

The CMCC adapter allows you to convert a dynamic link to a static link while the DLUR subsystem is running. Dynamic links are those links that are established outside of the scope of the TN3270 DLUR configuration. These links are either configured by the host or are established dynamically using the VRN function and are activated by DLUR or activated remotely.

There are several advantages of converting a dynamic link to a static link:

- Supports removing a DLUR link without having to shut down the entire DLUR subsystem.
- In Network Node server configurations, having two or three static links defined allows you to provide adequate redundancy. You might want to convert a dynamic link to a static link to provide this benefit.
- Static links allow better control from the router end to show and control them. Dynamic links cannot be specifically shown or controlled by the router. The links appear in **show** command output, but with locally assigned names such as @DLURnn which make them difficult to identify.

Converting a Dynamic Link to a Static Link

To convert a dynamic link to a static link the CMCC adapter allows you to re-enter the local/remote MAC/SAP quadruple in the **link** (TN3270) command, which the CMCC accepts as a request to convert the link to a static link, and does not reject the command due to a duplicate local/remote MAC/SAP quadruple.

For example, use the following **link** (TN3270) command to convert the existing dynamic link named HOST at RMAC 4000.0000.0001 and RSAP 4 to a static link:

```
link HOST rmap 4000.0000.0001 rsap 4
```

Removing a Dynamic Link

To remove a dynamic link use the following commands in DLUR SAP configuration mode to convert the dynamic link to a static link and then to remove the link:

	Command	Purpose
Step 1	`Router(tn3270-dlur-lsap)# link name [rmac rmac] [rsap rsap]`	Creates named links to hosts, or if this is an existing dynamic link, converts the dynamic link to a static link.
Step 1	`Router(tn3270-dlur-lsap)# no link name`	Removes the link definition.

Monitoring Dynamic LU Naming

To monitor the status of the Dynamic LU Naming enhancement, use the following commands in EXEC mode:

Command	Purpose
`Router# show extended channel tn3270-server`	Displays current server configuration parameters and the status of the PUs defined for the TN3270 server.
`Router# show extended channel tn3270-server pu client-name`	Displays configuration parameters for a PU and all the LUs currently attached to the PU, with the client machine name substituted for the client IP address.

Monitoring Inverse DNS Nailing

To monitor the status of the Inverse DNS Nailing enhancement, use the following commands in EXEC mode:

Command	Purpose
`Router# show extended channel tn3270-server client-name`	Displays information about all connected clients with a specific machine name.
`Router# show extended channel tn3270-server nailed-domain`	Lists all nailing statements with a specific nailed-domain name.
`Router# show extended channel tn3270-server nailed-name`	Lists all nailing statements with a specific nailed- machine name.
`Router# show extended channel tn3270-server pu client-name`	Displays configuration parameters for a PU and all the LUs currently attached to the PU, with the client machine name substituted for the client IP address.

Shutting Down the TN3270 Server and Its Entities

To shut down the entire TN3270 server or to shut down individual TN3270 server entities, use the **shutdown** command in the appropriate configuration mode. The **shutdown** command is available in multiple configuration modes, including interface configuration mode for the CMCC adapter. This support allows you to have varying levels of control for different configurable entities.

For TN3270 server configurations, you can use the **shutdown** command in the following command modes:

- TN3270 server configuration mode—Shuts down the entire TN3270 server function.
- PU configuration mode—Shuts down an individual PU entity within the TN3270 server.
- DLUR configuration mode—Shuts down the whole DLUR subsystem within the TN3270 server.
- DLUR PU configuration mode—Shuts down an individual PU within the SNA session switch configuration in the TN3270 server.
- DLUR SAP configuration mode—Shuts down the local SAP and its associated links within the SNA session switch configuration.

- Listen-point configuration mode—Shuts down a listen point and all of its associated configuration entities.
- Listen-point PU configuration mode—Shuts down an individual PU within the listen point configuration.

To shut down the TN3270 server or a specific entity within the TN3270 server configuration, use the following command in the appropriate configuration mode:

Command	Purpose
`Router# shutdown`	Shuts down the entities corresponding to the configuration level in which the **shutdown** command is entered.

TN3270 Server Configuration Examples

This section provides examples of router configurations for the TN3270 server. It provides LU pooling configuration examples with DLUR and with direct PU and legacy configuration examples without LU pooling:

Note The first three configuration examples in this section apply only to users who are already using TN3270.

Basic Configuration Example

The following example shows a router with a legacy TN3270 server configuration and PU specification prior to LU pooling and listen-point configuration support:

```
tn3270-server
 pu PU1 94223456 10.10.10.1 tok 1 08
  tcp-port 40
  keepalive 10
```

The following example shows the same router with a later TN3270 server configuration that replaces the existing configuration and uses the **listen-point** command to accomplish LU pooling. The **listen-point** command was first introduced in Cisco IOS Release 11.2(18)BC.

```
tn3270-server
```

```
 listen-point 10.10.10.1 tcp-port 40
   pu PU1 94223456 tok 1 08
    keepalive 10
```

Note In the new configuration, the IP address is not configured in the PU. Instead, the IP address is configured as a listen point and the PU is configured within the scope of the listen point. The **tcp-port** command is not configured within the scope of the PU, instead it is specified with the **listen-point** command.

Listen-Point Direct PU Configuration Example

The following example shows a router with a legacy TN3270 server configuration that contains different PUs configured with the same IP addresses:

```
tn3270-server
 pu PU1 94201231 10.10.10.2 tok 1 10
 pu PU2 94201232 10.10.10.3 tok 1 12
 pu PU3 94201234 10.10.10.3 tok 1 14
 pu PU4 94201235 10.10.10.4 tok 1 16
  tcp-port 40
 pu PU5 94201236 10.10.10.4 tok 2 08
```

The following example shows the same router replaced with a later TN3270 server configuration that uses the **listen-point** command introduced in Cisco IOS Release 11.2(18)BC:

```
tn3270-server
 listen-point 10.10.10.2
  pu PU1 94201231  tok 1 10
 listen-point 10.10.10.3
  pu PU2 94201232  tok 1 12
  pu PU3 94201234  tok 1 14
 listen-point 10.10.10.4
  pu PU5 94201236  tok 2 08
 listen-point 10.10.10.4 tcp-port 40
  pu PU4 94201235  tok 1 16
```

In this example, PU2 and PU3 are grouped into one listen point because they have the same IP address. Note that even though PU4's IP address is identical to PU5's IP address, they are not configured within the same listen point because the listen point indicates a unique IP address and TCP port pair. If you do not specify the TCP port, the default port value is 23.

Listen-Point DLUR PU Configuration Example

The following example shows a router with a legacy TN3270 server configuration for DLUR:

```
tn3270-server
 dlur NETA.RTR1 NETA.HOST
  dlus-backup NETA.HOST
  lsap token-adapter 15 08
  link MVS2TN   rmac 4000.b0ca.0016
  pu PU1 017ABCDE 10.10.10.6
```

The following example shows the same router replaced with a later TN3270 server configuration that uses the new **listen-point** command introduced in Cisco IOS Release 11.2(18)BC:

```
tn3270-server
 dlur NETA.RTR1 NETA.HOST
```

```
 dlus-backup NETA.HOST
 lsap token-adapter 15 08
 link MVS2TN   rmac 4000.b0ca.0016
listen-point 10.10.10.6
 pu PU1 017ABCDE dlur
```

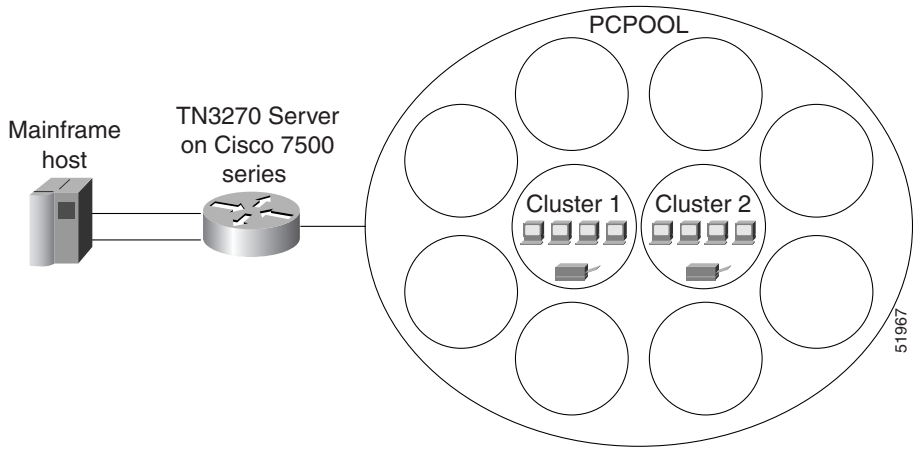
In this example, the PU is not configured within the scope of DLUR. Instead the PU is configured within the listen-point scope. The keyword **dlur** differentiates the listen-point direct PU from the listen-point DLUR PU. Note that the DLUR configuration must be completed before PU1 is configured.

Any siftdown commands configured within the scope of listen point are automatically inherited by the PUs that are configured within the scope of that listen point. To override the siftdown configurations, you can explicitly configure the siftdown configuration commands within the scope of the listen-point PU.

LU Pooling Configuration Example

Figure 280 shows a router running the TN3270 server (with DLUR PUs) and its LU pooling configuration.

Figure 280 TN3270 Server Using LU Pooling



To understand how LUs are allocated for clients that are nailed to pools in the TN3270 server, consider the router configuration for PU2 on the following pages, and assume that cluster 1 for PCPOOL has no LUs currently assigned to clients.

For a PC client with IP address 20.40.34.1, the TN3270 server reserves LUs 201–205 for cluster 1 of the PCPOOL. PCPOOL is defined with a cluster layout of “4s1p” for a total of 5 LUs (Figure 282). Because the cluster 1 LUs are reserved, a second PC client with IP address 20.40.34.7 (also nailed to the PCPOOL) is given LUs 206 to 210 for cluster 2 of the PCPOOL (provided that cluster 2 is the next available cluster without LUs currently allocated).

Next, consider that a total of 4 clients with IP address 20.40.34.1 have connected with a request for a screen LU. These clients are allocated LUs 201 to 204 (cluster 1) because according to the cluster definition “4s1p”, the first 4 LUs are screen LUs. According to the cluster definition the last (5th) LU is a printer LU.

This means that cluster 1 is fully allocated for screen LUs. In this example, the next client with IP address 20.40.34.1 that connects with a request for a screen LU reserves the next available cluster, with LUs 211 to 215. This client is allocated LU 211, which is a screen LU.

The first client with IP address 20.40.34.1 to request a printer LU from the TN3270 server is allocated LU 205. LU 205 is the first available printer LU in the first cluster of reserved LUs for IP address 20.40.34.1.

Clients that connect with a request for a specific pool but that are not nailed to that pool are allocated an LU from the generic pool. In this example, an available LU in the range 251 to 255 is allocated.

The following router configuration shows an example of commands used to define the TN3270 server with LU pools.

Router Configuration

```
logging buffered
! logs Cisco IOS software messages to the internal buffer using the default
! buffer size for the router platform
interface Channel 6/1
 no ip address
 no keepalive
 csna E160 40
!
interface Channel 6/2
 ip address 172.18.4.17 255.255.255.248
 no keepalive
 lan TokenRing 15
  source-bridge 15 1 500
  adapter 15 4000.b0ca.0015
 lan TokenRing 16
  source-bridge 16 1 500
  adapter 16 4000.b0ca.0016
 tn3270-server
  pool NEREGION cluster layout 1a
  pool PCPOOL   cluster layout 4s1p
  pool UNIXPOOL cluster layout 49s1p
  dlur NETA.SHEK NETA.MVSD
   lsap token-adapter 15 04
    link SHE1     rmac 4000.b0ca.0016
  listen-point 172.18.4.18
   client ip 10.20.20.30 pool UNIXPOOL
   client ip 10.20.40.0 255.255.255.0 pool PCPOOL
   client ip 10.20.30.0 255.255.255.128 pool NEREGION
   pu PU1      91903315 dlur
    allocate lu 1 pool PCPOOL   clusters 10
    allocate lu 51 pool UNIXPOOL clusters 2
    allocate lu 200 pool NEREGION clusters 50


  listen-point 172.18.4.19
   client ip 20.30.40.40 pool UNIXPOOL
   client ip 20.40.34.0 255.255.255.0 pool PCPOOL
   client ip 20.40.50.0 255.255.255.128 pool NEREGION
   pu PU2      91913315 dlur
    allocate lu 1 pool UNIXPOOL clusters 2
    allocate lu 101 pool NEREGION clusters 100
    allocate lu 201 pool PCPOOL   clusters 10
```

Figure 281 shows cluster layouts for PU1 in the TN3270 server.

Figure 281 Cluster Layouts for PU1 in the TN3270 Server

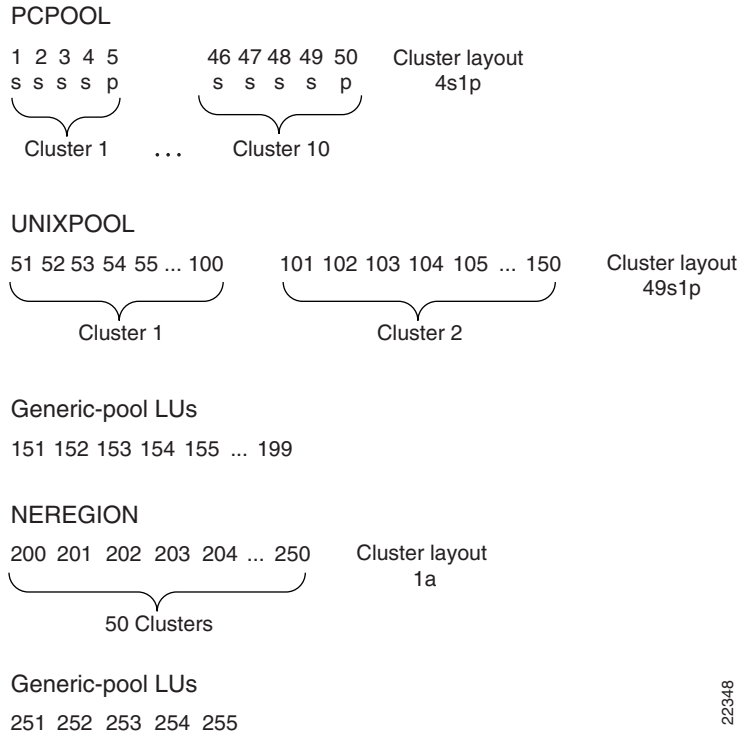
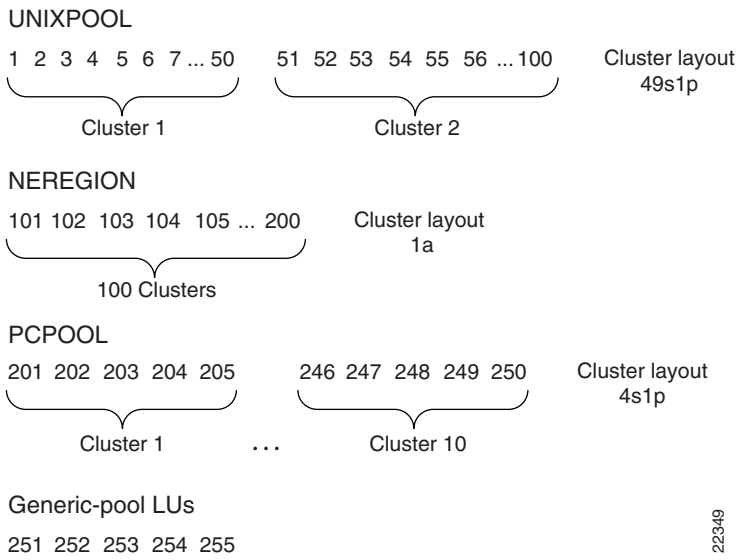


Figure 282 shows cluster layouts for PU2 in the TN3270 server.

Figure 282 Cluster Layouts for PU2 in the TN3270 Server



TN3270 Server Configuration Without LU Pooling Example

The following configuration shows three PUs using DLUR and two more with direct connections without LU pooling.

The initial CIP configuration is as follows:

```
interface Channel2/2
 ip address 10.10.20.126 255.255.255.128
 no ip redirects
 no ip directed-broadcast
 no keepalive
 lan TokenRing 0
  source-bridge 223 1 2099
  adapter 0 4100.cafe.0001
   llc2 N1 2057
  adapter 1 4100.cafe.0002
   llc2 N1 2057
```

Configuration dialog to configure the TN3270 function follows:

```
! HOSTA is channel-attached and will open SAP 8 on adapter 0.
! HOSTB is reached via token-ring
! HOSTC is channel-attached non-APPN and will open SAP 4 on adapter 0.

! enter interface configuration mode for the virtual interface in slot 2
router(config)#int channel 2/2

! create TN3270 Server entity
router(config-if)#tn3270-server

! set server-wide defaults for PU parameters
router(cfg-tn3270)#keepalive 0
router(cfg-tn3270)#unbind-action disconnect
router(cfg-tn3270)#generic-pool permit
```

```
! define DLUR parameters and enter DLUR configuration mode
router(cfg-tn3270)#dlur SYD.TN3020 SYD.VMG

! create a DLUR LSAP and enter DLUR LSAP configuration mode
router(tn3270-dlur-pu)#lsap token-adapter 1

! specify the VRN name of the network containing this lsap
router(tn3270-dlur-lsap)#vrn syd.lan4

! create a link from this lsap
router(tn3270-dlur-lsap)#link hosta rmac 4100.cafe.0001 rsap 8
router(tn3270-dlur-lsap)#link hostb rmac 4000.7470.0009 rsap 4
router(tn3270-dlur-lsap)#exit
router(tn3270-dlur)#exit

! create listen-points and DLUR PUs
router(cfg-tn3270)#listen-point 10.10.20.1
router(tn3270-lpoint)#pu pu0 05d99001 dlur
router(tn3270-lpoint-pu)#exit
router(tn3270-lpoint)#pu pu1 05d99002 dlur
router(tn3270-lpoint-pu)#exit
router(tn3270-lpoint)#exit

router(cfg-tn3270)#listen-point 10.10.20.2
router(tn3270-lpoint)#pu pu2 05d99003 dlur
router(tn3270-lpoint-pu)#exit
router(tn3270-lpoint)#exit

! create direct pus for the non-APPN Host
! note that they must use different lsaps because they go to the same Host

router(cfg-tn3270)#listen-point 10.10.20.5
router(tn3270-lpoint)#pu pu3 05d00001 tok 1 24 rmac 4100.cafe.0001 lu-seed pu3###
router(tn3270-lpoint-pu)#exit
router(tn3270-lpoint)#pu pu4 05d00002 tok 1 28 rmac 4100.cafe.0001 lu-seed pu4###
router(tn3270-lpoint-pu)#end
```

The following configuration results from the initial CIP configuration and the configuration dialog:

```
interface Channel2/2
 ip address 10.10.20.126 255.255.255.128
 no ip redirects
 no keepalive
 lan TokenRing 0
  source-bridge 223 1 2099
  adapter 0 4100.cafe.0001
   llc2 N1 2057
  adapter 1 4100.cafe.0002
   llc2 N1 2057
  tn3270-server
   dlur SYD.TN3020 SYD.VMG
    lsap token-adapter 1
     vrn SYD.LAN4
     link HOSTB    rmac 4000.7470.0009
     link HOSTA    rmac 4100.cafe.0001 rsap 08
   listen-point 10.10.20.1
    pu PU0 05D99001 dlur
    pu PU1 05D99002 dlur
   listen-point 10.10.20.2
    pu PU2 05D99003 dlur
   listen-point 10.10.20.5
    pu PU3 05D00001 tok 1 24 rmac 4100.cafe.0001 lu-seed PU3###
    pu PU4 05D00002 tok 1 28 rmac 4100.cafe.0001 lu-seed PU4###
```

TN3270 DLUR Configuration With CMPC Host Connection Example

The following example shows a DLUR PU with a CMPC host connection:

```
logging buffered
! logs Cisco IOS software messages to the internal buffer using the default
! buffer size for the router platform
interface Channel0/0
 no ip address
 no keepalive
 cmpc C010 E5 LPAR1TG READ
 cmpc C010 E6 LPAR1TG WRITE
 cmpc C020 00 LPAR2TG READ
 cmpc C020 01 LPAR2TG WRITE
!
interface Channel0/2
 ip address 172.18.5.1 255.255.255.224
 no keepalive
 lan TokenRing 0
  source-bridge 100 1 8
  adapter 0 4000.4040.0000 ! for cmpc
  adapter 1 4000.6060.0000 ! TN3270 server
  adapter 2 4000.7070.0000
 tn3270-server
  maximum-lus 20000 ! optional
  idle-time 64800 ! optional
  timing-mark ! optional
  tcp-port 24 ! optional
  client 10.10.10.0 255.255.255.0 lu maximum 10000 ! optional


  dlur NETA.TN3270CP NETA.CPAC
   dlus-backup NETA.MVS2 ! optional
   preferred-NNserver NETA.CPAC ! optional
   lsap token-adapter 1 04 ! TN3270 server uses cmcc adapter 1 and sap=04
    link LINK1 rmac 4000.4040.0000 rsap 08 ! link to cmpc on adapter 0
   lsap token-adapter 2 04
    link LINK2 rmac 4000.7070.0000 rsap 08 ! link to cmpc on adapter 2
   listen-point 172.18.5.2
    pu TNPU1 01754321 dlur
!
tg LPAR1TG llc token-adapter 0 08 rmac 4000.6060.0000 rsap 04 ! rsap optional
tg LPAR2TG llc token-adapter 2 08 rmac 4000.7070.0000 ! rsap=04 by default"
```

Removing LU Nailing Definitions Example

In the following example, locaddrs 1 to 50 are reserved for all remote screen devices in the 171.69.176.0 subnet:

```
interface channel 2/2
  tn3270-server
  pu BAGE4
  client ip 171.69.176.28 255.255.255.0 lu 1 50
```

To remove a nailing definition, the complete range of LOCADDRS must be specified as configured. So for the example above, the following command would remove the LU nailing definition:

```
no client ip 171.69.176.28 255.255.255.0 lu 1 50
```

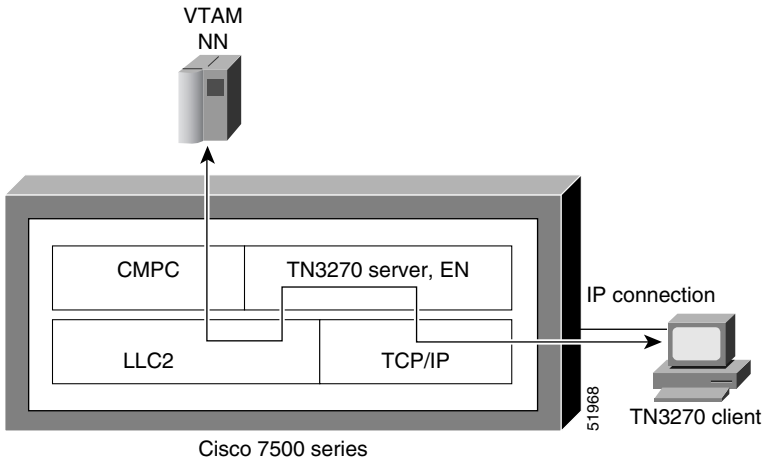
If an attempt is made to remove a subset of the range of configured LOCADDRS then the command is rejected:

```
no client ip 171.69.176.28 255.255.255.0 lu 1 20
% client ip 171.69.176.28 lu not matched with configured lu 1 50
```

TN3270 Server DLUR Using CMPC Example

Figure 283 shows the physical components for this example. Figure 284 shows the various parameters for each component in the configuration example.

Figure 283 Topology for VTAM-to-TN3270 Server DLUR Using CMPC

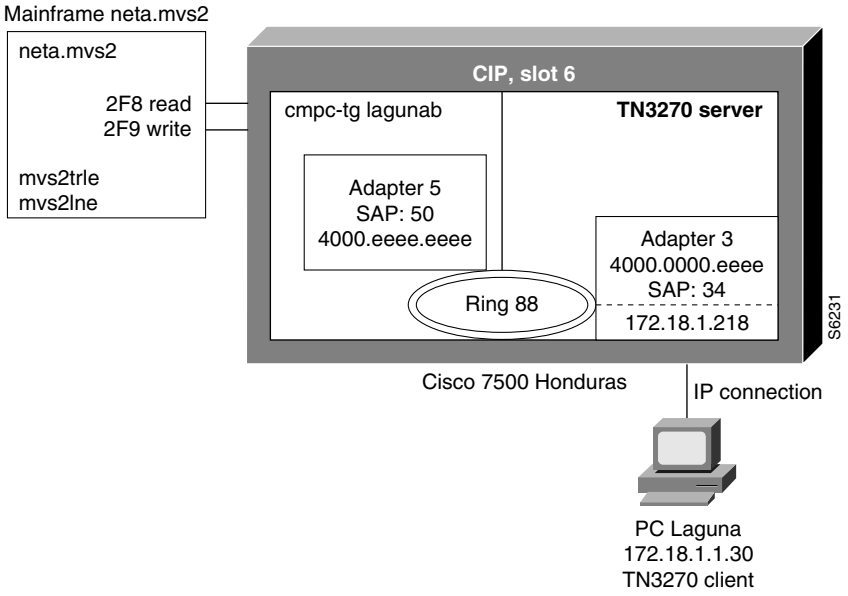


In Figure 283, the following activity occurs:

- The TN3270 server on the CMCC adapter takes on the role of an APPN EN running DLUR.
- The APPN NN in VTAM communicates with the CMPC driver over the channel.
- The CMPC driver on the CMCC adapter passes the data to the LLC2 stack on the CIP via a fast-path loopback driver to the TN3270 server on the CIP.
- The TN3270 server converts the 3270 data stream to a TN3270 data stream and forwards the packets to the IP TN3270 clients in the IP network.

The TN3270 server does not have to be in the same CMCC adapter as the CMPC driver.

Figure 284 Parameters for VTAM-to-TN3270 DLUR Using CMPC



The following configurations apply to the example shown in Figure 284.

mvs2trle

```
MVS2TRE  VBUILD TYPE=TRL
 MVS2TRLE TRLE  LNCTL=MPC,MAXBFRU=8,REPLYTO=3.0,
                READ=(2F8),
                WRITE=(2F9)
```

mvs2lne

```
MVS2NNE  VBUILD TYPE=LOCAL
MVS2PUE  PU    TRLE=MVS2TRLE,
               ISTATUS=ACTIVE,
               XID=YES,CONNTYPE=APPN,CPCP=YES
```

swlagtn

```
SWLAGTN  VBUILD TYPE=SWNET,MAXGRP=10,MAXNO=10,MAXDLUR=10
LAGTNPU PU     ADDR=01,                                          X
               MAXPATH=1,                                        X
               IDBLK=017,IDNUM=EFEED,                            X
               PUTYPE=2,                                         X
               MAXDATA=4096,                                     X
               LUGROUP=TNGRP1,LUSEED=LAGLU##
```

tngrp1

```
TNGRP1E  VBUILD TYPE=LUGROUP
TNGRP1   LUGROUP
DYNAMIC  LU    DLOGMOD=D4C32XX3,                                 X
               MODETAB=ISTINCLM,USSTAB=USSTCPIP,SSCPFM=USS3270
@        LU    DLOGMOD=D4C32784,                                 X
               MODETAB=ISTINCLM,USSTAB=USSTCPIP,SSCPFM=USS3270
```

Additional Router Configuration for Router Honduras

```
logging buffered
! logs Cisco IOS software messages to the internal buffer using the default
```

```
! buffer size for the router platform
interface Channel6/1
 cmpc C020 F8 CONFIGE READ
 cmpc C020 F9 CONFIGE WRITE
!
interface Channel6/2
 lan TokenRing 0
  source-bridge 88 3 100
  adapter 5 4000.eeee.eeee
  adapter 6 4000.0000.eeee
 tn3270-server
  dlur NETA.HOND327S NETA.MVS2
   lsap token-adapter 6  54
    link MVS2TN   rmac 4000.eeee.eeee rsap 50
   listen-point 172.18.1.218
    pu TNPU 017EFEED dlur
 tg CONFIGE  llc token-adapter 6 50 rmac 4000.eeee.eeee rsap 54
```

Activate the Configuration

On the MVS system, use the following commands to activate the configuration:

```
v net,act,id=mvstrle,update=add
v net,act,id=mvslne
v net,act,id=swhondpu
v net,act,id=swlagtn
v net,act,id=swhondcp
v net,act,id=tngrp1
```

Dynamic LU Naming Example

Router configuration

The following router configuration is an example of the TN3270 server configured with LU pooling. A listen-point PU is configured to define DLUR PUs using dynamic LU naming. Note the following lines in the configuration:

- The **lu deletion** command must be configured with the **named** option.
- The PU pu1 is defined with lu-seed abc##pqr. Using hexadecimal numbers for *##*, the LU names for this PU are ABC01PQR, ABC02PQR, ABC03PQR.... up to ABCFFPQR. Similarly, the PU pu2 is defined with lu-seed pqr###. Using decimal numbers for *###*, the LU names for this PU are PQR001, PQR002... up to PQR255.

The LUs ABC01PQR through ABC32PQR and PQR100 through PQR199 are allocated to the pool SIMPLE. The LUs ABC64PQR through ABC96PQR and PQR010 through PQR035 are allocated to the pool PCPOOL. The remaining LUs are in the generic pool.

```
tn3270-server
 pool simple cluster layout 1s
 pool pcpool cluster layout 4s1p
 lu deletion named
 dlur neta.shek neta.mvsd
  lsap tok 15 04
    link she1 rmac 4000.b0ca.0016
 listen-point 172.18.4.18
 pu pu1 91903315 tok 16 08 lu-seed abc##pqr
!
!The following statement allocates LUs ABC01PQR through ABC32PQR to the pool named
!simple.
!
  allocate lu 1 pool simple clusters 50
```

```
!
!The following statement allocates LUs ABC64PQR through ABC96PQR to the pool named
!pcpool.
!
  allocate lu 100 pool pcpool clusters 10
 pu pu2 91913315 dlur lu-seed pqr###
!
!The following statement allocates LUs PQR010 through PQR035 to the pool named pcpool.
!
  allocate lu 10 pool pcpool clusters 5
!
!The following statement allocates LUs PQR100 through PQR199 to the pool named simple.
!
  allocate lu 100 pool simple clusters 100
```

Mainframe configuration

The following mainframe configuration is an example of the VTAM configuration that can be used if the TN3270 server is configured with the Dynamic LU Naming enhancement.

Note PUs are defined with the LUGROUP command. It is not necessary to specify an LUSEED. If the LUSEED operand is specified, it is ignored.

Note You must specify the INCLUD0E=YES parameter on VTAM so that the TN3270 server receives the LU name generated by the VTAM exit.

```
SWN72022 VBUILD TYPE=SWNET
PU1      PU     ADDR=01,                                          X
                PUTYPE=2,                                         X
                IDBLK=919,                                        X
                IDNUM=03315,                                      X
                INCLUD0E=YES,                                     X
                LUGROUP=MYLUS
*
PU2      PU     ADDR=01,                                          X
                PUTYPE=2,                                         X
                IDBLK=919,                                        X
                IDNUM=13315,                                      X
                INCLUD0E=YES,                                     X
                LUGROUP=MYLUS
```

Inverse DNS Nailing Examples

Nailing Clients to Pools by Device Name, Domain Name, and Domain ID using a Domain ID

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing:

```
tn3270-server
  domain-id 2 .cisco.com
  domain-id 20 .yahoo.com
  pool GENERAL  cluster layout 4s1p
  pool TEST  cluster layout 4s1p
  listen-point 172.18.5.168
   pu T240CA   91922363 token-adapter 31 12 rmac 4000.4000.0001
    allocate lu 1 pool GENERAL  clusters 1
   client name lucy49.cisco.com pool GENERAL
   client name george 20 pool TEST
```

```
   client name arthur 20 pool TEST
   client name tyson 20 pool TEST
   client name daisy 20 pool TEST
  listen-point 172.18.5.169
   pu T240CB   91922364 token-adapter 31 12 rmac 4000.4000.0002
    allocate lu 1 pool TEST     clusters 50
   client domain-name cisco.com pool GENERAL
   client domain-id 20 pool TEST
```

Nailing Clients to Pools by IP Address

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing. In this example, the **client pool** command is configured with the **ip** keyword. The command nails the client at IP address 10.1.2.3 with an IP mask of 255.255.255.0 to the pool named OMAHA:

```
tn3270-server
 pool OMAHA cluster layout 10s1p
 listen-point 172.18.4.18
 client ip 10.1.2.3 255.255.255.0 pool OMAHA
```

Nailing Clients to Pools by Device Name

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing. In this example the **client pool** command is configured with the **name** keyword. The command nails the client at device name george-isdn29.cisco.com to the pool named GENERAL:

```
tn3270-server
  pool GENERAL  cluster layout 4s1p
  listen-point 172.18.5.168
   pu T240CA   91922363 token-adapter 31 12 rmac 4000.4000.0001
    allocate lu 1 pool GENERAL  clusters 1
  client name george-isdn29.cisco.com pool GENERAL
```

Nailing Clients to Pools by Device Name using a Domain ID

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing. In this example the **client pool** command is configured with the **name** keyword and the optional *DNS-domain-identifier* argument. The command nails the client at device named lucy-isdn49.cisco.com to the pool named GENERAL:

```
tn3270-server
 domain-id 23 .cisco.com
  pool GENERAL  cluster layout 4s1p
  listen-point 172.18.5.168
   pu T240CA   91922363 token-adapter 31 12 rmac 4000.4000.0001
    allocate lu 1 pool GENERAL  clusters 1
 client name lucy-isdn49 23 pool GENERAL
```

Nailing Clients to Pools by Domain Name

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing. In this example the **client pool** command is configured with the **domain**-**name** keyword. The command nails any client at domain name .cisco.com to the pool named GENERAL:

```
tn3270-server
  pool GENERAL  cluster layout 4s1p
  listen-point 172.18.5.168
   pu T240CA   91922363 token-adapter 31 12 rmac 4000.4000.0001
    allocate lu 1 pool GENERAL  clusters 1
 client domain-name .cisco.com pool GENERAL
```

Nailing Clients to Pools by Domain Name Using a Domain ID

The following router configuration shows an example of commands used to define the TN3270 server with LU pools using inverse DNS nailing. In this example the **client pool** command is configured with the **domain-id** keyword. The command nails any client at domain name .cisco.com to the pool named GENERAL:

```
tn3270-server
 domain-id 23 .cisco.com
  pool GENERAL  cluster layout 4s1p
  listen-point 172.18.5.168
   pu T240CA   91922363 token-adapter 31 12 rmac 4000.4000.0001
    allocate lu 1 pool GENERAL  clusters 1
 client domain-id 23 pool GENERAL
```

SSL Encryption Support Examples

Mainframe configuration

The following mainframe configuration is an example of the VTAM configuration that can be used if the SSL Encryption Support enhancement is configured:

```
example PU definition:
*
BMPU4    PU     ADDR=01,
                PUTYPE=2,
                LOGAPPL=NETTMVSD,
                LUGROUP=BMCL13,LUSEED=BMPU4###,
                PACING=8,VPACING=8,
                IDBLK=919,
                IDNUM=36821
*
BMPU5    PU     ADDR=01,
PUTYPE=2,
                LOGAPPL=NETTMVSD,
                LUGROUP=BMCL13,LUSEED=BMPU5###,
                PACING=8,VPACING=8,
                IDBLK=919,
                IDNUM=46821
*
BMPU6    PU     ADDR=01,
                PUTYPE=2,
                LOGAPPL=NETTMVSD,
                USSTAB=USSTCPMF,
                DLOGMOD=D4C32782,
                PACING=8,VPACING=8,
                IDBLK=919,
                IDNUM=56821
*
BMPU6001 LU    LOCADDR=01
BMPU6002 LU    LOCADDR=02
BMPU6003 LU    LOCADDR=03
BMPU6004 LU    LOCADDR=04
BMPU6005 LU    LOCADDR=05
BMPU6006 LU    LOCADDR=06
BMPU6007 LU    LOCADDR=07
BMPU6008 LU    LOCADDR=08
BMPU6009 LU    LOCADDR=09
BMPU6010 LU    LOCADDR=10
.
BMPU6255 LU    LOCADDR=255
*
```

Simple SSL Encryption Support Example

The following router configuration shows an example of commands used to define a simple configuration of the SSL Encryption Support enhancement. In this configuration, listen-point 172.18.5.187 is a secured listen-point using security profile cert40. Note that the security profile is using all of the default parameters.

```
interface Channel3/2
 ip address 172.18.5.185 255.255.255.248
 no keepalive
 lan TokenRing 15
  source-bridge 15 1 500
  adapter 15 4000.b0ca.0015
 lan TokenRing 16
  source-bridge 16 1 500
  adapter 16 4000.b0ca.0016
 tn3270-server
  security
   profile CERT40 SSL
    servercert slot0:verisign187.pem
  listen-point 172.18.5.187
   sec-profile CERT40
   pu BMPU5    91946821 token-adapter 15 08 rmac 4000.b0ca.0016
```

Complex SSL Encryption Support Example

The following router configuration shows an example of commands used to define a more complex configuration of the SSL Encryption Support enhancement:

- Listen-point 172.18.5.186 is a non-secured listen point.
- Listen-point 172.18.5.187 is a secured listen-point using security-profile cert128 with the encryption order specified and a keylen of 128 which implies strong (domestic) encryption.
- Listen-point 172.18.5.188 is a secured listen-point using security profile cert40 with default security-profile parameters.

```
interface Channel3/2
 ip address 172.18.5.185 255.255.255.248
 no keepalive
 lan TokenRing 15
  source-bridge 15 1 500
  adapter 15 4000.b0ca.0015
 lan TokenRing 16
  source-bridge 16 1 500
  adapter 16 4000.b0ca.0016
```

```
tn3270-server
 security
  profile CERT128 SSL
   servercert slot0:verisign128.pem
   encryptorder RC4 RC2 DES
   keylen 128
  profile CERT40 SSL
   servercert slot0:coach188.pem
 listen-point 172.18.5.186
  pu BMPU4    91946821 token-adapter 15 04 rmac 4000.b0ca.0016
 listen-point 172.18.5.187
  sec-profile CERT128
  pu BMPU5    91956821 token-adapter 15 08 rmac 4000.b0ca.0016
 listen-point 172.18.5.188
  sec-profile CERT40
  pu BMPU6    91966821 token-adapter 15 0C rmac 4000.b0ca.0016
```