

## Configuring X.25 and LAPB

---

This chapter describes how to configure connections through X.25 networks, including Link Access Procedure, Balanced (LAPB) connections. LAPB procedures are presented first for those users who only want to configure a simple, reliable serial encapsulation method. For a complete description of the commands mentioned in this chapter, refer to the “X.25 and LAPB Commands” chapter in the *Router Products Command Reference* publication. For historical background and a technical overview of X.25 and LAPB, see the *Internetworking Technology Overview* publication. To translate between X.25 and another protocol, refer to the *Protocol Translator Configuration Guide and Command Reference* publication.

### Cisco’s Implementation of LAPB and X.25

A group of specifications published by the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) defines *Recommendations*, or specifications, for X.25. The ITU-T specifications basically specify connections between data terminal equipment (DTE) and data communications equipment (DCE) for remote terminal access and computer communications. The X.25 specifications include LAPB as the data link layer protocol and X.25 as the network layer protocol, or packet layer protocol (PLP) as it is also known. The ITU-T updates its specifications every four years, and the specifications dated 1980 and 1984 are the most common versions currently in use. Additionally, the International Standards Organization (ISO) has published ISO 7776:1986 as an equivalent to the LAPB standard, and ISO 8208:1989 as an equivalent to the ITU-T 1984 X.25 Recommendation packet layer. Our X.25 software follows the ITU-T 1984 X.25 Recommendation, except for its Defense Data Network (DDN) and Blacker Front End (BFE) operation, which follow the ITU-T 1980 X.25 Recommendation.

---

**Note** The ITU-T carries out the functions of the former Consultative Committee for International Telegraph and Telephone (CCITT).

---

In addition to providing remote terminal access, our X.25 software provides transport for LAN protocols—IP, DECnet, XNS, ISO CLNS, AppleTalk, Novell IPX, Banyan VINES, and Apollo Domain—and bridging. For information about these protocols, refer to the specific protocol chapters in the *Router Products Configuration Guide*.

Briefly, the Cisco Systems X.25 software provides the following capabilities:

- LAPB datagram transport—LAPB is a protocol that operates at Level 2 (the data link layer) of the OSI reference model. It offers a reliable connection service for exchanging data (in units called frames). Protocol datagrams (IP, DECnet, AppleTalk, and so forth) are carried over a reliable LAPB connection, or datagrams of several of these protocols are encapsulated in a proprietary protocol and carried over a LAPB connection. The LAPB connection is configured to carry a single protocol or multiple protocols.
- X.25 datagram transport—Protocol datagrams (IP, DECnet, AppleTalk, and so forth) are encapsulated inside packets on an X.25 virtual circuit. Mappings between X.25 addresses and protocol addresses allow these datagrams to be routed through an X.25 network, thereby allowing an X.25 public data network (PDN) to transport LAN protocols.
- X.25 switch—X.25 calls can be routed based on their X.25 addresses either between serial interfaces on the same router (local switching) or across an IP network to another router (remote switching, also called *tunneling*). Remote X.25 switching encapsulates the X.25 packet level inside a TCP connection, allowing X.25 equipment to be connected via a TCP/IP-based network.
- Connection-Mode Network Service (CMNS)—CMNS is a mechanism through which local X.25 switching can be extended to different media (Ethernet, FDDI, and Token Ring) by using OSI-based NSAP addresses. This implementation provides X.25 over LLC2 to allow X.25 over media other than serial interfaces. Cisco's CMNS implementation supports services defined in ISO Standards 8208 (packet level) and 8802-2 (frame level).
- DDN and BFE X.25—DDN-specified Standard Service is supported. The DDN X.25 Standard Service is the required protocol for use with DDN Packet-Switched Nodes (PSNs). The Defense Communications Agency (DCA) has certified Cisco Systems' DDN X.25 Standard Service implementation for attachment to the Defense Data Network. Cisco's DDN implementation also includes Blacker Front End and Blacker Emergency Mode operation.
- X.25 MIB—Specified in SNMP MIB Extension for X.25 LAPB (RFC 1381) and SNMP MIB Extension for the X.25 Packet Layer (RFC 1382) subsets are supported. The LAPB XID Table X.25 Cleared Circuit Table, and X.25 Call Parameter Table are not implemented. All values are read-only. To use the X.25 MIB, refer to the *Cisco Management Information Base (MIB) User Quick Reference* publication, or the RFCs.
- X.25 support—Our software supports all mandatory requirements of 1988 X.25.

Our X.25 implementation does not support fast switching.

## Facility Handling in Encapsulated X.25 Virtual Circuits

The router either originates or accepts encapsulation switched virtual circuits (SVCs) in order to transport LAN traffic through an X.25 network.

When the router originates a Call for LAN traffic encapsulation, the facilities in the Call are controlled by the facilities configured for the interface and the map statement that specifies the LAN/X.25 encapsulation. Because a router can be attached to a Public Data Network (PDN), the interface and map configurations allow a number of facilities to be specified in outgoing Calls. These facilities are specified in all originated Calls relating to the given interface and map with one exception; the incoming and outgoing maximum packet sizes proposed will be lowered if the lower layer (LAPB) cannot support the specified DATA packet size.

When the router accepts an encapsulation Call, many facilities are simply ignored. The maximum packet sizes will be lowered if the lower layer (LAPB) cannot support the sizes proposed. A reverse-charged Call will be Cleared if neither the interface nor the map allows it. A Call that specifies a Network User Identification (NUID) will be Cleared if the user authentication fails.

## Facility Handling in Routed X.25 Virtual Circuits

Routed X.25 traffic might have facilities added, deleted, or modified by the router.

### Standard (1984 X.25) Facilities

Table 11-1 describes how standard (1984 X.25) facilities are treated when routing a switched virtual circuit (SVC). To configure these facilities, refer to the "Configure X.25 User Facilities" section.

**Table 11-1 Treatment of Standard X.25 Facilities**

Facility	Treatment
Flow Control Negotiation (negotiation of window size and maximum packet size)	The router adds, removes, or changes flow control parameter values to match the values on both interfaces, as described in the following cases.
<ul style="list-style-type: none"> <li>Requested flow control values do not match the outgoing interface's defaults.</li> <li>Requested values match the outgoing interface's defaults.</li> <li>Requested maximum packet size exceeds the capability of either interface.</li> <li>Call is routed from a modulo 128 interface to a modulo 8 interface.</li> <li>Call is remotely routed over a TCP connection.</li> <li>Call is received from an X.25 over TCP connection without one or more flow control parameter values.</li> <li>Accepted flow control parameter values are different, for any reason, from the values proposed for the incoming Call.</li> </ul>	<ul style="list-style-type: none"> <li>Router inserts flow control parameters into the outgoing switched Call.</li> <li>Router strips parameter values from the outgoing switched Call.</li> <li>Router lowers the packet size to the largest value that can be supported by the two interfaces.</li> <li>Router lowers the larger requested window size to 7.</li> <li>Router X.25 code ensures that both proposed maximum packet sizes and proposed window sizes for a Call are present.</li> <li>By default, the router forces the Call to use the maximum packet sizes (128/128) and window sizes (2/2). If the <b>x25 routing use-tcp-if-defs</b> command and keyword are specified, the router will assume that the Call takes the default values of the outgoing serial interface. In either case, the Call Confirm sent back over the X.25-over-TCP (XOT) connection will indicate the final flow control values negotiated for the connection.</li> <li>Router sends an outgoing Call Accepted packet that indicates the accepted flow control values.</li> </ul>
Throughput Negotiation	Router forwards the incoming Throughput facility.
Closed User Group Selection	Router forwards a basic format Closed User Group selection facility; any other format of Closed User Group selection (extended format, CUG with outgoing access or Bilateral CUG) will be stripped.
Reverse Charging	Router forwards an incoming Reverse Charging facility.
Fast Select	Router forwards an incoming Fast Select facility.
Network User Identification (NUID)	Router forwards an incoming NUID facility on a Call packet; an NUID facility on a Call Accepted packet will be stripped.
Charging Information	Router strips any Charging Information or Request.
RPOA Selection	Router strips any RPOA Selection.

Facility	Treatment
Called Line Address Modified Notification	Router forwards a Called Line Address Modified Notification.
Call Redirection Notification	Router strips a Call Redirection Notification.
Transit Delay Selection	Router forwards an incoming Transit Delay facility.

The implementation of X.25 prior to software Release 9.1(4.1) did not insert flow control parameter values into Call packets sent over X.25-over-TCP (XOT) connections. When such an XOT Call is received by software Release 9.1(4.1) or later, the Call will be forced to the standard flow control values. This may cause migration problems when the router is connecting X.25 equipment that is not capable of negotiating flow control parameters; the optional **use-tcp-if-defs** keyword of the **x25 routing** command can be used if this problem is encountered.

### ITU-T-Specified Marker Facilities

Table 11-2 describes how ITU-T-specified marker facilities are treated when routing an SVC.

**Table 11-2 Default Treatment of ITU-T-Specified Marker Facilities**

Facility	Treatment
Calling Address Extension	Router forwards an incoming Calling Address Extension facility.
Called Address Extension	Router forwards an incoming Called Address Extension facility.
Quality of Service Negotiation	Router strips any of the Quality of Service facilities.
Expedited Data Negotiation	Router strips an Expedited Data Negotiation facility.

The router requires the Calling Address Extension to route to a CMNS host.

The encoding of any CCITT/ITU-T facilities is preceded by a marker, as displayed by the output of the **debug x25** command.

## Local Marker Facilities Specified for DDN or BFE X.25

Table 11-3 describes how local marker facilities are treated when routing an SVC.

**Table 11-3 Default Treatment of Local Marker Facilities Specified for DDN or BFE X.25**

Facility	Treatment
DDN Service Type	Router strips an incoming DDN Service Type facility from a Call, but inserts DDN Service Type if a forwarded Call Accepted packet specifies a DDN precedence facility.
DDN Precedence	Router forwards an incoming DDN Precedence facility. However, both the input and output interfaces need to be configured for DDN X.25 encapsulation. To configure treatment of this facility, see the “Define IP Precedence Handling” section.
BFE Emergency Mode Addressing	Router strips an incoming BFE Emergency Mode Addressing facility. To configure treatment of this facility, see the “Configure Blacker Front-End X.25” section.

Our routers support DDN Standard service but not DDN Basic service. Consequently, DDN Service Type does not have to be configured.

## LAPB Configuration Task List

It is possible to use only LAPB as a serial encapsulation method. This can be done using a leased serial line. You must use one of the X.25 packet-level encapsulations when attaching to an X.25 network.

The LAPB standards distinguish between two types of hosts: data terminal equipment (DTE), and data circuit-terminating equipment (DCE). At Level 2, or the data link layer in the OSI model, LAPB allows for orderly and reliable exchange of data between a DTE and a DCE. A router using LAPB encapsulation can act as a DTE or DCE device at the protocol level, which is distinct from the hardware DTE or DCE level.

Using LAPB under noisy conditions can result in greater throughput than HDLC encapsulation. When LAPB detects a missing frame, the router retransmits the frame instead of waiting for the higher layers to recover the lost information. This behavior is good only if the host timers are relatively slow. In the case of quickly expiring host timers, however, you will discover that LAPB is spending much of its time transmitting host retransmissions. If the line is not noisy, the lower overhead of HDLC encapsulation is more efficient than LAPB. When using long delay satellite links, for example, the lock-step behavior of LAPB makes HDLC encapsulation the better choice.

To configure LAPB, complete the tasks in the following sections. The tasks in the first section are required; the remaining are optional.

- Establish LAPB DCE or DTE Operation
- Configure LAPB Timers and Frame Parameters
- Define the LAPB Hold Queue Size
- Monitor and Maintain LAPB and X.25

Examples of LAPB configurations are given at the end of this chapter.

## Establish LAPB DCE or DTE Operation

Set the appropriate LAPB encapsulation to run datagrams over a serial interface. One end of the link must be DTE and the other must be DCE.

Task	Command
Specify a serial interface.	<b>interface serial</b> <i>type number</i> <sup>1</sup>

1. This command is documented in the “Interface Commands” chapter in the *Router Products Command Reference* publication.

To select an encapsulation and the protocol if using a single protocol, or to select the multiple protocol operation, perform one or more of the following tasks in interface configuration mode:

Task	Command
Enable encapsulation of a single protocol on the line using DCE operation.	<b>encapsulation lapb dce</b>
Enable encapsulation of a single protocol on the line using DTE operation.	<b>encapsulation lapb</b>
Set the line protocol.	<b>lapb protocol</b> <i>protocol</i>
Enable use of multiple protocols on the line using DCE operation.	<b>encapsulation lapb dce multi</b>
Enable use of multiple protocols on the line using DTE operation.	<b>encapsulation lapb multi</b> <sup>1</sup>

1. Multi-LAPB does not support bridging (SRB and transparent) or TCP header compression. Only protocols supported by LAPB encapsulation are supported by multiprotocol LAPB encapsulation.

For an example of configuring LAPB DCE operation, see the section “Typical LAPB Configuration Example” later in this chapter.

## Configure LAPB Timers and Frame Parameters

X.25 Level 2 or LAPB operates at the data link layer of the OSI reference model. LAPB specifies methods for exchanging data (in units called *frames*), detecting out-of-sequence or missing frames, retransmitting frames, and acknowledging frames.

The LAPB modulo determines the operating mode. Modulo 8 (basic mode) is widely available, because it is required for all standard LAPB implementations and is sufficient for most links. Modulo 128 (extended mode) can achieve greater throughput on high-speed links that have a low error rate (satellite links, for example) by increasing the number of frames that can be transmitted before waiting for acknowledgment (as configured by the LAPB window parameter, *k*). By its design, LAPB’s *k* parameter can be at most one less than the operating modulo. Modulo 8 links can typically send seven frames before an acknowledgment must be received; modulo 128 links can set *k* to a value as large as 127. By default, LAPB links use the basic mode with a window of 7.

When connecting to an X.25 network, use the N1 parameter value set by the network administrator. This value is the maximum number of bits in a LAPB frame, which determines the maximum size of an X.25 packet. When using LAPB over leased lines, the N1 parameter should be eight times the hardware maximum transmission unit (MTU) size plus any LAPB overhead. The other frame parameters are determined by the network configuration; see Table 11-4 for the default values.

The retransmission timer (T1) determines how long a transmitted frame can remain unacknowledged before the router polls for an acknowledgment. For X.25 networks, the router retransmission timer setting should match that of the network.

The LAPB standards define a timer to detect unsignaled link failures (T4). The T4 timer is reset every time a frame is received from the partner on the link. If the T4 timer expires, a Receiver Ready frame with the Poll bit set is sent to the partner, which is required to respond. If the partner does not respond, the standard polling mechanism is used to determine whether the link is down. The period of T4 must be greater than the period of T1.

Another LAPB timer function allows brief hardware failures, while the protocol is up, without requiring a protocol reset. If a brief hardware outage occurs, the link will continue uninterrupted if the outage is cured before the specified hardware outage period expires.

For leased-line circuits, the T1 timer setting is critical because the design of LAPB assumes that a frame has been lost if it is not acknowledged within period T1. The timer setting must be large enough to permit a maximum-sized frame to complete one round trip on the link. If the timer setting is too small, the router will poll before the acknowledgment frame can return, which may result in duplicated frames and severe protocol problems. If the timer setting is too large, the router waits longer than necessary before requesting an acknowledgment, which reduces bandwidth.

Table 11-4 summarizes the LAPB parameters you can set in interface configuration mode.

**Table 11-4 LAPB Parameters**

Task (LAPB Parameter)	Command	Values or Ranges	Default
Set the modulo.	<b>lapb modulo</b> <i>modulus</i>	8 or 128	8
Set the window size (k).	<b>lapb k</b> <i>window-size</i>	1– (modulo minus 1) frames	7
Set maximum bits per frame (N1).	<b>lapb n1</b> <i>bits</i>	1088–32840 bits (must be a multiple of 8)	Based on hardware MTU and protocol overhead
Set count for sending frames (N2).	<b>lapb n2</b> <i>tries</i>	1–255 tries	20
Set the retransmission timer (T1).	<b>lapb t1</b> <i>milliseconds</i>	1–64000 milliseconds	3000
Set the hardware outage period.	<b>lapb interface-outage</b> <i>milliseconds</i>		0 (disabled)
Set the idle link period (T4).	<b>lapb t4</b> <i>seconds</i>		0 (disabled)

For an example of configuring the LAPB T1 timer, see the section “Typical LAPB Configuration Example” later in this chapter.

## Define the LAPB Hold Queue Size

To define the maximum number of frames to be held while the other side is not accepting data, perform the following task in interface configuration mode:

Task	Command
Define the size of a frame hold queue.	<b>lapb hold-queue</b> <i>queue-size</i>

## X.25 Configuration Task List

To configure X.25, complete the tasks in one or more of the following sections, depending upon the X.25 application or task required for your network:

- Configure an X.25 Interface
- Configure an X.25 Datagram Transport
- Configure X.25 Routing
- Configure CMNS Routing
- Configure DDN or BFE X.25
- Configure X.25 Level 3 Optional Parameters and Special Features
- Monitor and Maintain LAPB and X.25

Note that all X.25 applications except DDN and BFE X.25 operation and routing require that an encapsulation method; it might be optional or required to set an interface X.121 address, depending on what you expect the router to do and on the network circumstances. See the sections “Set the Interface X.25 Mode” and “Set the Interface X.121 Address” for the tasks to set them.

Default parameters have been provided for X.25 operation; however, you can change the settings to meet the needs of your X.25 network or as defined by your X.25 service supplier. We also provide special and custom configuration settings to further optimize your X.25 usage. See the section “Configure X.25 Level 3 Optional Parameters and Special Features” later in this chapter for more information.

See the end of this chapter for examples of configuring X.25.

### Configure an X.25 Interface

To configure an X.25 interface on the router, perform the tasks in the following sections:

- Set the Interface X.25 Mode
- Set the Interface X.121 Address
- Set the Default Flow Control Values

The first task is required. The others might be required or optional, depending on what the router is expected to do and on the circumstances of your network.

#### Set the Interface X.25 Mode

A router using X.25 Level 3 encapsulation can act as a DTE or DCE protocol device (according to the needs of your X.25 service supplier), can use DDN or BFE encapsulation, or can use the IETF standard encapsulation, as specified by RFC 1356.

Because the default serial encapsulation is HDLC, you must explicitly configure an X.25 encapsulation method.

To configure the mode of operation and one of these encapsulation types for a specified interface, perform the following task in interface configuration mode:

Task	Command
Set X.25 mode of operation.	<code>encapsulation x25 [dte   dce] [[ddn   bfe]   [ietf]]</code>



Typically a public data network will require attachment as a DTE. (This is distinct from the hardware interface DTE and DCE assignments.)

The default mode of operation is DTE, and the default encapsulation method is Cisco's pre-IETF method. If either DDN or BFE operation is needed, it must be explicitly configured.

For an example of configuring X.25 DTE operation, see the section "Typical X.25 Configuration Example" later in this chapter.

## Set the Interface X.121 Address

If your router does not originate or terminate Calls but only participates in X.25 switching, this task is optional. However, if the router is attached to a PDN, you must set an interface X.121 address.

Any attachment to a PDN should set the X.121 address value that is assigned by the X.25 network service provider. To set the X.121 address, perform the following task in interface configuration mode:

Task	Command
Set the X.121 address.	<b>x25 address</b> <i>x.121-address</i>

For an example of configuring the X.25 interface address, see the section "Typical X.25 Configuration Example" later in this chapter.

## Set the Default Flow Control Values

Setting correct default flow control parameters (window size and packet size) is essential for correct operation of the link because X.25 is a strongly flow controlled protocol. However, it is easy to overlook this task because many networks use standard default values.

To configure flow control parameters, complete the tasks in the following sections. These tasks are optional if your X.25 attachment uses the standard default values for maximum packet sizes (128 bytes incoming and outgoing) and window sizes (two packets incoming and outgoing).

- Set default window sizes.
- Set default packet sizes.

### Set Default Window Sizes

X.25 networks have a default input and output window size that is defined by the network administrator. You must set the router default input and output window sizes to match those of the network; see the note following the next section, "Set Default Packet Sizes." To set the default window sizes (default is 2), perform the following tasks in interface configuration mode:

Task	Command
Set the default number of packets the server can receive before sending an X.25 acknowledgment.	<b>x25 win</b> <i>packets</i>
Set the default number of sent packets that can remain unacknowledged before the virtual circuit uses its hold queue.	<b>x25 wout</b> <i>packets</i>

For an example of setting the default window sizes, see the sections “Typical X.25 Configuration Example” and “DDN X.25 Configuration Example” later in this chapter.

### Set Default Packet Sizes

X.25 networks have a default maximum input and output packet size (default is 128) that is defined by the network administrator. To set the router default input and output maximum packet sizes to match those of the network, perform the following tasks in interface configuration mode:

Task	Command
Set the default input maximum packet size.	<b>x25 ips bytes</b>
Set the default output maximum packet size.	<b>x25 ops bytes</b>

To send a packet larger than the agreed X.25 packet size over an X.25 virtual circuit, a router must break the packet into two or more X.25 packets with the M-bit (“more data” bit) set. The receiving device collects all packets in the M-bit sequence and reassembles them into the original packet.

It is possible to define default packet sizes that cannot be supported by the lower layer (see the LAPB N1 parameter). The router will, however, negotiate lower maximum packet sizes for all SVCs so the agreed sizes can be carried.

For an example of setting the default maximum packet sizes, see the sections “Typical X.25 Configuration Example” and “DDN X.25 Configuration Example” later in this chapter.

---

**Note** Because the X.25 protocol requires the DTE and DCE to have identical default maximum packet sizes and default window sizes, if the interface is up, changes to the window and packet sizes will be held until the X.25 protocol RESTARTs the packet service.

---

## Configure an X.25 Datagram Transport

X.25 support is most commonly configured as a transport for datagrams across an X.25 network. This is accomplished by first establishing a mapping between protocol addresses (for example, IP or DECnet) and the X.121 addresses of the X.25 network. When datagrams for a particular destination are routed for the first time, a virtual circuit is set up to the appropriate X.121 address. The Call User Data portion of the initial Call Request identifies the protocol of the datagrams being carried by a particular virtual circuit.

Figure 11-1 illustrates two routers sending data across an X.25 public data network (PDN).



**Figure 11-1 Transporting LAN Protocols across an X.25 PDN**

Perform the tasks in the following sections, as necessary, to complete the X.25 configuration for your network needs:

- Map Protocol Addresses to X.121 Addresses
- Configure X.25 Payload Compression
- Establish a PVC
- Establish a Default Virtual Circuit Protocol
- Configure Subinterfaces

The following sections describe how to perform these configuration tasks. Configuring the X.25 parameters and special features, including TCP header compression and X.25 bridging, are described in the section “Configure X.25 Level 3 Optional Parameters and Special Features” later in this chapter.

## Map Protocol Addresses to X.121 Addresses

This section describes the X.25 single-protocol and multiprotocol encapsulation options that are now available, describes how protocols are identified under both options, and describes how to map protocol addresses to an X.121 address for a remote host.

### Protocol Encapsulation for Single-Protocol and Multiprotocol Virtual Circuits

We have long supported encapsulation of a number of datagram protocols across X.25, using a standard method when available, or a proprietary method when necessary. These traditional methods assign a protocol to each virtual circuit. If more than one protocol is carried between the router and a given host, each active protocol will have at least one virtual circuit dedicated to carrying its datagrams.

We also support a newer standard, RFC 1356, which standardizes a method for encapsulating most datagram protocols over X.25. It also specifies how one virtual circuit can carry datagrams from more than one protocol.

A router can be configured to use any of the available encapsulation methods with a particular host.

Once an encapsulation virtual circuit is established using any method, sending and receiving a datagram is a simple process of fragmenting and reassembling the datagram into and from an X.25 complete packet sequence. An X.25 complete packet sequence is one or more X.25 data packets that have the More bit set in all but the last packet. A virtual circuit that can carry multiple protocols includes protocol identification data as well as the protocol data at the start of each complete packet sequence.

### Protocol Identification for Single-Protocol and Multiprotocol Virtual Circuits

The primary difference between the available Cisco and IETF encapsulation methods is the specific value used to identify a protocol. When any of the methods establishes a virtual circuit for carrying a single protocol, the protocol is identified in the call packet by using the Call User Data (CUD) field. When a virtual circuit is established to carry more than one protocol (only available using the RFC 1356 methodology), a specific value is also encoded in the Call User Data field in each datagram; every datagram exchanged over that virtual circuit has its protocol identified.

Table 11-5 summarizes the values used in the Call User Data field to identify protocols.

**Table 11-5 Protocol Identification in the Call User Data Field**

Protocol	Cisco Protocol Identifier	IETF RFC 1356 Protocol Identifier
Apollo Domain	0xD4	0x80 (5-byte SNAP encoding <sup>1</sup> )
AppleTalk	0xD2	0x80 (5-byte SNAP encoding)
Banyan VINES	0xC0 00 80 C4 <sup>2</sup>	0x80 (5-byte SNAP encoding)
Bridging	0xD5	(Not implemented)
ISO CLNS	0x81	0x81 <sup>3</sup>
Compressed TCP	0xD8	0x00 (5-byte SNAP encoding) <sup>4</sup>
DECnet	0xD0	0x80 (5-byte SNAP encoding)
IP	0xCC	0xCC <sup>5</sup> or 0x80 (5-byte SNAP encoding)
Novell IPX	0xD3	0x80 (5-byte SNAP encoding)
QLLC	0xC3	(Not available)
XNS	0xD1	0x80 (5-byte SNAP encoding)
Multiprotocol	(Not available)	0x00

1. SNAP encoding is defined from the Assigned Numbers RFC; Cisco's implementation recognizes only the IETF OUI 0x00 00 00 followed by a two-byte Ethernet protocol type.
2. The use of 0xC0 00 80 C4 for Banyan VINES is defined by Banyan.
3. The use of 0x81 for CLNS is compatible with ISO/IEC 8473-3:1994.
4. Compressed TCP traffic has two types of datagrams, so IETF encapsulation requires a multiprotocol virtual circuit.
5. The use of 0xCC for IP is backwards-compatible with RFC 877.

Once a multiprotocol virtual circuit has been established, datagrams on the virtual circuit have protocol identification data before the actual protocol data; the protocol identification values are the same used by RFC 1356 in the CUD field for an individual protocol.

---

**Note** IP datagrams can be identified using a 1-byte identification (0xCC) or a 6-byte identification (0x80 followed by the 5-byte SNAP encoding). The 1-byte encoding is used by default, although the SNAP encoding can be configured.

---

### Map Datagram Addresses to X.25 Hosts

Encapsulation is a cooperative process between the router and another X.25 host. Since X.25 hosts are reached using an X.121 address (an X.121 address has between 0 and 15 decimal digits), the router must have a means to map a host's protocols and addresses to its X.121 address.

Each encapsulating X.25 interface should be configured with the relevant datagram parameters. For example, an interface that encapsulates IP will typically have an IP address.

A router set up for DDN or BFE service uses a dynamic mapping technique to convert between IP and X.121 addresses. These techniques have been designed specifically for attachment to the DDN network and to Blacker encryption equipment. Their design, restrictions, and operation make them work well for these specific applications, but not for other networks.

You should also establish the X.121 address of an encapsulating X.25 interface using the **x25 address** interface configuration command. This is the address that encapsulation calls should be directed to. It will also be used as the source X.121 address when originating an encapsulation call, which is how the destination host will be able to determine what services are needed for the encapsulation virtual circuit. A DDN or BFE interface will have an X.121 address generated from the interface IP address, which for proper operation, should not be modified.

For each X.25 interface, you can explicitly map each destination host's protocols and addresses to its X.121 address. If needed and the destination host has the capability, one host map can be configured to support several protocols; alternatively, you can define one map for each supported protocol.

To establish a map, perform the following task in interface configuration mode:

Task	Command
Map one or more host protocol addresses to the host's X.121 address.	<b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9] x.121-address [option]</i>

As an example, if you are encapsulating IP over a given X.25 interface, you should define an IP address for the interface and, for each of the desired destination hosts, map the host's IP address to its X.121 address.

---

**Note** You can map an X.121 address to as many as nine protocol addresses, but each protocol can be mapped only once in the command line.

---

An individual host map can use the given keyword to specify the following protocols:

- **apollo**—Apollo Domain
- **appletalk**—AppleTalk
- **bridge**—Bridging
- **clns**—OSI Connectionless Network Service
- **compressedtcp**—TCP header compression
- **decnet**—DECnet
- **ip**—IP
- **ipx**—Novell IPX
- **qllc**—IBM's QLLC
- **vines**—Banyan VINES
- **xns**—XNS

A multiprotocol map can specify a single address for all of the protocols listed above (except bridging and QLLC, which are not supported for RFC 1356 encapsulation). If IP and TCP header compression are both specified, the same IP address must be given for both protocols.

When setting up the address map, you can include options, such as enabling broadcasts and specifying the number of virtual circuits allowed, and defining various user facility settings.

---

**Note** Multiprotocol maps, especially those configured to carry broadcast traffic, can result in significantly larger traffic loads, requiring a larger hold queue, larger window sizes, or multiple virtual circuits.

---

For specific information about how to establish a protocol to run over X.25, refer to the appropriate protocol chapters in this publication or in the *Router Products Command Reference* publication.

The configuration for the Open Shortest Path First (OSPF) protocol can be greatly simplified by adding the optional **broadcast** keyword. See the **x25 map** command description in the “X.25 and LAPB Commands” chapter of the *Router Products Command Reference* publication for more information.

## Configure X.25 Payload Compression

For increased efficiency on relatively slow networks, our routers support X.25 payload compression of outgoing encapsulation traffic.

Several restrictions apply to X.25 payload compression:

- The compressed virtual circuit must connect two Cisco routers, because X.25 payload compression is not standardized.

The data packets conform to the X.25 protocol rules, so a compressed virtual circuit can be switched through standard X.25 equipment, but only Cisco routers can compress and decompress the data.

- Only datagram traffic can be compressed, although all of the encapsulation methods supported by Cisco routers are available (for example, an IETF multiprotocol virtual circuit can be compressed).

Switched virtual circuits can not be translated between compressed and non-compressed data, nor can PAD data be compressed.

- X.25 payload compression should be applied carefully.

Each compressed virtual circuit requires significant memory resources (for a dictionary of learned data patterns) and computation resources (every data packet received is decompressed and every data packet sent is compressed). Excessive use of compression can cause unacceptable overall router performance.

- X.25 compression must be explicitly configured for a map command.

A received Call that specifies compression will be rejected if the corresponding host map does not specify the “compress” option. An incoming Call that does not specify compression can, however, be accepted by a map that specifies compression.

To enable payload compression over X.25, perform the following task in interface configuration mode:

Task	Command
Enable payload compression over X.25.	<b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9] x.121-address compress</i>

This command specifies that X.25 compression is to be used when mapping outgoing traffic to the specified host. Because each virtual circuit established for compressed traffic uses significant amounts of memory, compression should be used with careful consideration of its impact on the router's performance.

## Establish a PVC

Permanent virtual circuits (PVCs) are the X.25 equivalent of leased lines; they are never disconnected. You must specify the required network protocol-to-X.121 address map before you set up a PVC.

To establish a PVC, perform the following task in interface configuration mode:

Task	Command
Set a encapsulation PVC.	<b>x25 pvc</b> <i>circuit protocol address [protocol2 address2...protocol9 address9] x.121-address [option]</i>

See the previous section for a list of protocols; all but CMNS are supported by this task. For an example of configuring a PVC, see the section "Using a PVC to Exchange IP Traffic Example" later in this chapter.

## Establish a Default Virtual Circuit Protocol

The Call Request packet that sets up a virtual circuit might contain a field called the Call User Data (CUD) field. Typically, the software uses the first byte of Call User Data to distinguish which high-level protocol will be carried by a particular virtual circuit.

Table 11-6 lists the hexadecimal values of the Call User Data and its corresponding network level protocol. The use of 0x81 for ISO 8473 (CLNS) and 0x01 for PAD are ISO standards. The use of 0xCC for IP is defined by RFC 877, and is recognized by ISO. The use of C0 00 80 C4 is defined by Banyan. The other values are meaningful only to our X.25 software. Most of the 1-byte identifiers are padded to 4 bytes with 3 bytes of 0x00. BFE IP encapsulation requires that only 1 byte be used. CLNS may use 1 or 5 bytes as defined in ISO 8473.

**Table 11-6 Protocols and Initial Byte of Call User Data**

Protocol	Initial CUD Byte
PAD	0x01
ISO CLNS	0x81
Banyan VINES	0xC0 00 80 C4
DOD IP	0xCC
DECnet	0xD0
XNS	0xD1

Protocol	Initial CUD Byte
AppleTalk	0xD2
Novell IPX	0xD3
Apollo Domain	0xD4
Bridging	0xD5
TCP Header Compression (THC)	0xD8

You can specify that calls with unknown CUD or with no CUD are to be interpreted as IP or PAD calls by performing the following task in interface configuration mode:

Task	Command
Establish the default protocol.	<b>x25 default</b> <i>protocol</i>

## Configure Subinterfaces

Subinterfaces are virtual interfaces that can be used to connect several networks to each other through a single physical interface. Subinterfaces are made available on our routers because routing protocols, especially those using the split horizon principle, may need help to determine which hosts need a routing update. The split horizon principle, which allows routing updates to be distributed to other routed interfaces except the interface on which the routing update was received, works well in a LAN environment in which other routers reached by the interface have already received the routing update.

However, in a WAN environment using connection-oriented interfaces (like X.25 and Frame Relay), other routers reached by the same physical interface might not have received the routing update. Rather than forcing network administrators to connect routers by separate physical interfaces, we provide subinterfaces that are treated as separate interfaces. A network administrator can separate hosts into subinterfaces of a physical interface, the X.25 protocol is unaffected, and routing processes see each subinterface as a separate source of routing updates, so all subinterfaces are eligible to receive routing updates.

## Point-to-Point and Multipoint Subinterfaces

There are two types of subinterfaces: point-to-point and multipoint. Subinterfaces are implicitly multipoint, unless configured as point-to-point.

A point-to-point interface is used to connect two hosts for a given protocol. An X.25 point-to-point subinterface will accept only a single encapsulation command (such as **x25 map** or **x25 pvc**) for a given protocol (though you can use multiple encapsulation commands, one for each protocol), so there can be only one destination for the protocol. All protocol traffic routed to a point-to-point interface will be forwarded to the one destination host defined for the interface. (Because only one destination is defined for the interface, the routing process does not even have to consult the destination address in the datagrams.)

A multipoint subinterface is used to connect one or more hosts for a given protocol. There is no restriction on the number of encapsulation commands that can be configured on a multipoint subinterface. Because the hosts appear on the same subinterface, they are not relying on the router to distribute routing updates between them. When a routing process forwards a datagram to a multipoint subinterface, the X.25 encapsulation process must be able to map the datagram's destination address to a configured encapsulation command. If the routing process cannot find a map for the datagram destination address, the encapsulation will fail.



---

**Note** Because of the complex operations dependent on a subinterface and its type, the router will not allow a subinterface's type to be changed, nor can a subinterface with the same number be established again once it has been deleted. Once a subinterface has been deleted, it takes a reload to remove all internal references. However, the deleted subinterface can be easily reconstituted using a different subinterface number.

---

## Creating and Configuring X.25 Subinterfaces

To create and configure a subinterface, complete the following tasks in interface configuration mode:

Task	Command
Create a point-to-point or multipoint subinterface.	<b>interface serial</b> <i>number.subinterface-number</i> <b>[point-to-point   multipoint]</b> <sup>1</sup>
Configure X.25 encapsulation for each protocol to be sent on the interface	<b>x25 map</b> <i>protocol address x.121-address [option]</i>
or	
Establish an encapsulation PVC for a protocol on the subinterface or both.	<b>x25 pvc</b> <i>circuit protocol address x.121-address [option]</i>

1. This command is documented in the "Interface Commands" chapter of the *Router Products Command Reference* publication.

For an example of configuring an X.25 subinterface and using multiple encapsulation commands for a single destination address, see the "Creating and Configuring a Point-to-Point Subinterface Example" section later in this chapter. For more general information about configuring subinterfaces, refer to the "Configuring Interfaces" chapter of this manual.

---

**Note** When configuring IP routing over X.25, you might need to make adjustments to accommodate split horizon effects. Refer to the "Configuring IP Routing Protocols" chapter of this manual for details about how the router handles possible split horizon conflicts. By default, split horizon is enabled for X.25 networks.

---

## Configure X.25 Routing

The X.25 software implementation allows virtual circuits to be routed from one X.25 interface to another and from one router to another. The routing behavior can be controlled with switching and tunneling commands, based on a locally built table.

Encapsulated LAN protocols can share an X.25 serial interface with the X.25 switching support. Switching or forwarding X.25 virtual circuits can be done two ways:

- Incoming calls received from a local serial interface running X.25 can be forwarded to another local serial interface running X.25. This is known as *local X.25 switching* because the router handles the complete path. It does not matter whether the interfaces are configured as DTE or DCE devices, because the software will take the appropriate actions.
- An incoming call also can be forwarded to another of our routers over a LAN using the TCP/IP protocols. Upon receipt of an incoming call, a TCP stream connection is established to the router that is acting as the switch for the destination. All X.25 packets are sent and received over this

reliable data stream. Flow control is maintained end-to-end. This is known as *remote X.25 switching*, or *tunneling*. It does not matter whether the interfaces are configured as DTE or DCE, because the software will take the appropriate actions.

Running X.25 over TCP/IP provides a number of benefits. The datagram containing the X.25 packet can be switched by other routers using their high-speed switching abilities. X.25 connections can be sent over networks running only the TCP/IP protocols. The TCP/IP protocol suite runs over many different networking technologies, including Ethernet, Token Ring, T1 serial, and FDDI. Thus X.25 data can be forwarded over these media to another router, where it can be output to an X.25 interface.

When the connection is made locally, the switching configuration is used; when the connection is across a LAN, the tunneling configuration is used. The basic function is the same for both types of connections, but different configuration commands are required for the two types of connections.

The X.25 switching subsystem supports the following facilities and parameters:

- The D-bit ignored but passed through transparently
- Variable-length interrupt data (if not operating as a DDN or BFE interface)
- Flow Control Parameter Negotiation
  - Window size up to 7, or 127 for modulo 128 operation
  - Packet size up to 4096 (if the LAPB layers used are capable of handling the requested size)
- Basic Closed User Group selection
- Throughput class negotiation
- Reverse charging and fast select

The tasks for configuring these facilities are described in the section “Configure X.25 Level 3 Optional Parameters and Special Features.”

To configure X.25 tunneling on your router, perform the following tasks:

- Enable X.25 Routing
- Construct the X.25 Routing Table
- Translate X.25 Called and Calling Addresses
- Update the X.121 Address
- Configure a (Local) Switched PVC
- Configure a (Remote) Tunneled PVC

You also need to configure the X.25 parameters and special features, as required for your network. Each task is described in a following section.

## Enable X.25 Routing

You can enable switching, or tunneling, (used for remote switching), allowing you to route X.25 traffic through a LAN. To enable local X.25 routing, perform the following task in global configuration mode:

Task	Command
Enable X.25 routing.	<code>x25 routing [use-tcp-if-defs]</code>

The **USE-TCP-IF-DEFS** flag is used by some routers that receive remote routed calls from older software versions; it might be needed if the originating router cannot be migrated to a new software release.

For an example of configuring X.25 routing, see the sections “X.25 Route Address Pattern Matching Example” and “X.25 Routing Example” later in this chapter.

## Construct the X.25 Routing Table

The X.25 routing table is consulted when an incoming call is received that should be forwarded to its destination. Two fields are used to determine the route: the called X.121 network interface address (or destination host address) and the X.25 packet’s CUD field. When the destination address and the CUD of the incoming packet fit the X.121 and CUD patterns in the routing table, the call is forwarded. Additionally, when local switching, you can translate called and calling addresses. To construct an X.25 routing table, you must perform one of the following tasks in global configuration mode:

Task	Command
Construct the X.25 routing table by interface type.	<b>x25 route</b> [#position] x121-address [cud pattern] <b>interface</b> interface number
or	
Construct the X.25 routing table by IP address (tunneling).	<b>x25 route</b> [#position] x121-address [cud pattern] <b>ip</b> ip-address [ip-address2...ip-address6]

Entries have as a destination an interface (local switching) or IP address (remote switching). Up to six remote IP addresses can be entered as alternate routes that will be tried in turn to get a successful connection.

You can also specify an XOT source that causes the XOT TCP connection to use the IP address of a specified interface as the source address of the TCP connection. If, for instance, a loopback interface is specified for the XOT connection’s source address, TCP can use a primary interface or any backup interface to reach the other end of the connection. However, if a physical interface’s address is specified as the source address, the XOT connection is terminated if that interface goes down.

To configure an XOT route (thus adding it to the X.25 routing table), perform the following task in global configuration mode:

Task	Command
Configure an XOT (remote) X.25 route (tunneling).	<b>x25 route</b> [#position] x121-address [cud pattern] <b>ip</b> ip-address [xot-source type number]

**Note** It can take up to 50 seconds to try an alternate route due to TCP timings.

For an example of constructing the routing table, see the section “X.25 Routing Example” later in this chapter.

## Translate X.25 Called and Calling Addresses

When interconnecting two separate X.25 networks, it is sometimes necessary to provide for address translation. Your X.25 switch supports translation of X.25 called and calling addresses for local switching. To translate addresses, perform the following task in global configuration mode:

Task	Command
Translate X.25 called and calling addresses when switching.	<b>x25 route</b> [#position] x121-address [ <b>substitute-source</b> rewrite-pattern] [ <b>substitute-dest</b> rewrite-pattern] [ <b>cud pattern</b> ] <b>interface</b> type number

Note that address substitution is only performed on routes to an interface. When running X.25 over IP, address substitution can be performed on the destination IP system if the destination system is configured with the appropriate X.25 routing commands.

## Update the X.121 Address

Some X.25 calls, when forwarded by the X.25 switching support, need the calling (source) X.121 address updated to that of the outgoing interface. This is necessary when forwarding calls from private data networks to public data networks; other networks might require either address to be suppressed.

To update or suppress the X.121 address, perform one of the following tasks in interface configuration mode:

Task	Command
Update the X.121 address.	<b>x25 use-source-address</b>
Suppress the calling (source) X.121 address in Call Request packets.	<b>x25 suppress-calling-address</b>
Suppress the called (destination) X.121 address in Call Request packets.	<b>x25 suppress-called-address</b>

## Configure a (Local) Switched PVC

You can configure an X.25 PVC in the X.25 switching software. This means that DTEs that require permanent circuits can be connected to a router acting as an X.25 switch and have a properly functioning connection. X.25 RESETs will be sent to indicate when the circuit comes up or goes down. Both interfaces must define complementary locally switched PVCs.

To configure a locally switched PVC, perform the following task in interface configuration mode:

Task	Command
Configure a switched PVC.	<b>x25 pvc</b> number1 <b>interface</b> type number <b>pvc</b> number2 [option]

For an example of configuring a local switched PVC, see the section “Switching a PVC on the Same Router Example” later in this chapter.

## Configure a (Remote) Tunneled PVC

A PVC can be forwarded to another router over a LAN using the TCP/IP protocols. When the interfaces come up, a TCP stream connection is established to the router that is acting as the switch for the destination. All X.25 packets will be sent and received over this reliable data stream. Flow control is maintained end-to-end. This is known as *remote PVC switching* or *tunneling*.

Running X.25 over TCP/IP provides a number of benefits. Other routers can switch IP datagrams containing the X.25 packets using the router's high-speed switching abilities. X.25 data can be sent over networks running only TCP/IP protocols. The TCP/IP protocol suite runs over many different networking technologies, including Ethernet, Token Ring, T1 serial, and FDDI. Thus X.25 data can be forwarded over these media to another of our routers where it can be output to an X.25 interface. Both interfaces must define complementary tunneled PVCs.

To connect a PVC across a TCP/IP LAN, perform the following task in interface configuration mode:

Task	Command
Connect a PVC across a TCP/IP LAN.	<b>x25 pvc number1 tunnel address interface serial string pvc number2 [option]</b>

For an example of configuring a remote tunneled PVC, see the section "Establishing a Connection between Two PVCs Example" later in this chapter.

## Configure CMNS Routing

The Connection-Mode Network Service (CMNS) provides a mechanism through which local X.25 switching can be extended to nonserial media by using OSI-based NSAP addresses. This implementation runs packet-level X.25 over frame-level LLC2.

---

**Note** For information about configuring LLC2 parameters, refer to the "Configuring LLC2 and SDLC Parameters" chapter in the *Router Products Configuration Guide*.

---

In addition, our CMNS implementation allows LAN-based OSI resources, such as a DTE host and a Sun workstation, to be interconnected to each other via the router's LAN interfaces *and* to a remote OSI-based DTE through a WAN interface (using, for example, an X.25 PSN).

---

**Note** CMNS is implicitly enabled whenever an X.25 encapsulation is included with a serial interface configuration.

---

All local mapping is performed by statically mapping MAC addresses and X.121 addresses to NSAP addresses.

Implementing CMNS routing involves completing the tasks in the following sections:

- Enable CMNS on an Interface
- Specify a CMNS Static Map of Addresses

## Enable CMNS on an Interface

Enable CMNS on a nonserial interface by performing the following task in interface configuration mode:

Task	Command
Enable CMNS.	<b>cmns enable</b>

For an example of enabling CMNS on an interface, see the section “Enabling CMNS for X.121 and MAC Addresses Example” later in this chapter.

## Specify a CMNS Static Map of Addresses

After enabling CMNS on a nonserial interface (or specifying X.25 encapsulation on a serial interface), you must map NSAP addresses to either MAC-layer addresses or X.121 addresses, depending on the application.

For CMNS support over dedicated serial links (such as leased lines), an X.121 address is not needed, but can be optionally included. You must specify the X.121 address for CMNS connections over a packet-switched network, and you must specify a MAC address for CMNS connections over a nonserial medium (Ethernet, FDDI, or Token Ring).

To map the NSAP addresses to either a MAC address or X.121 address, perform one of the following tasks in interface configuration mode:

Task	Command
Statically map NSAP address to non-serial MAC-layer address.	<b>x25 map cmns nsap mac-address</b>
Statically map NSAP address to X.25, with an optional X.121 destination address.	<b>x25 map cmns nsap x.121-address</b>

For an example of configuring a CMNS static map of addresses, see the section “Enabling CMNS for X.121 and MAC Addresses Example” later in this chapter.

## Configure DDN or BFE X.25

The DDN X.25 protocol has two versions: Basic Service and Standard Service. Our X.25 implementation only supports the Standard Service. DDN X.25 Standard Service requires that the X.25 data packets carry IP datagrams. The DDN Packet Switch Nodes (PSNs) can extract the IP datagram from within the X.25 packet and pass data to another Standard Service host.

The DDN X.25 Standard is the required protocol for use with DDN PSNs. The Defense Communications Agency (DCA) has certified our DDN X.25 Standard implementation for attachment to the Defense Data Network. As part of the certification, our software is required to provide a scheme for dynamically mapping Internet addresses to X.121 addresses. See the section “DDN X.25 Dynamic Mapping” that follows for details on that scheme.

To enable DDN X.25 service, perform the tasks in the following sections:

- Enable DDN X.25
- Define IP Precedence Handling

To enable BFE X.25 service, perform the task in the following section:

- Configure Blacker Front-End X.25

## DDN X.25 Dynamic Mapping

The DDN X.25 standard implementation includes a scheme for dynamically mapping all classes of IP addresses to X.121 addresses without a table. This scheme requires that the IP and X.121 addresses conform to the formats shown in Figures 11-2 and 11-3. These formats segment the IP addresses into network (N), host (H), logical address (L), and PSN (P) portions. For the BFE encapsulation, the IP address is segmented into Port (P), Domain (D), and BFE ID number (B). The DDN algorithm requires that the host value be less than 64.

Class A:	Net.Host.LH.PSN → 0000 0 PPPHH00
Bits:	8 8 8 8
Class B:	Net.Net.Host.PSN → 0000 0 PPPHH00
Bits:	8 8 8 8
Class C:	Net.Net.Net.Host.PSN → 0000 0 PPPHH00
Bits:	8 8 8 4 4

S2302

Figure 11-2 DDN IP Address Conventions

BFE Class A :	Net.unused.Port.Domain.BFE → 0000 0 PDDDBBB
Bits:	8 1 3 10 10

S2823

Figure 11-3 BFE IP Address Conventions

## BFE IP Address Conventions

The DDN conversion scheme uses the host and PSN portions of an IP address to create the corresponding X.121 address. The DDN conversion mechanism is limited to Class A IP addresses; however, the router can convert Class B and Class C addresses as well. As indicated, this method uses the last two octets of a Class B address as the host and PSN identifiers, and the upper and lower four bits in the last octet of a Class C address as the host and PSN identifiers, respectively. The BFE conversion scheme requires a Class A IP address.

The DDN conversion scheme uses a physical address mapping if the host identifier is numerically less than 64. (This limit derives from the fact that a PSN cannot support more than 64 nodes.) If the host identifier is numerically larger than 64, the resulting X.121 address is called a logical address. The DDN does not use logical addresses.

The format of physical DDN X.25/X.121 addresses is ZZZZFIIHHZZ(SS), where each character represents a digit. ZZZZ represents four zeros, F is zero to indicate a physical address, III represents the PSN octet from the IP address padded with leading zeros, HH is the host octet from the IP address padded with leading zeros, and ZZ represents two zeros. (SS) represents the optional and unused subaddress.

The physical and logical mappings of the DDN conversion scheme always generate a 12-digit X.121 address. Subaddresses are optional; when added to this scheme, the result is a 14-digit X.121 address. The DDN does not use subaddressing.

Packets using routing and other protocols that require broadcast support can successfully traverse X.25 networks, including the DDN. This traversal requires the use of network protocol-to-X.121 maps, because the router must know explicitly where to deliver broadcast datagrams. (X.25 does not support broadcasts.) You can mark network protocol-to-X.121 map entries to accept broadcast packets; the router then sends broadcast packets to hosts with marked entries. For DDN or BFE operation, the router generates the interface X.121 addresses from the interface IP address using the DDN or BFE mapping technique.

## Enable DDN X.25

Both DCE and DTE operation cause the router to specify the Standard Service facility in the Call Request packet, which notifies the PSNs to use Standard Service.

To enable DDN X.25, perform one of the following tasks in interface configuration mode, as appropriate for your network:

Task	Command
Set DDN X.25 DTE operation.	<b>encapsulation x25 ddn</b>
Set DDN X.25 DCE operation.	<b>encapsulation x25 dce ddn</b>

For an example of enabling DDN X.25, see the section “DDN X.25 Configuration Example” later in this chapter.

## Define IP Precedence Handling

Using Standard Service, the DDN can be configured to provide separate service for datagrams with high precedence values. If the router receives an IP packet with a nonzero Internet precedence field, it uses a different virtual circuit which requested the DDN-specified precedence mapping in the Call Request packet. Different virtual circuits are maintained based on the precedence mapping values and the number of virtual circuits permitted.

By default, the DDN X.25 software opens one virtual circuit for all types of service values. You can enable the precedence sensitivity feature by performing the following task in interface configuration mode:

Task	Command
Allow a new virtual circuit based on the type of service (TOS) field.	<b>x25 ip-precedence</b>

Some hosts send nonstandard data in the TOS field, thereby causing multiple, wasteful virtual circuits to be created.

## Configure Blacker Front-End X.25

For environments that require a high level of security, your router software supports attachment to Defense Data Network Blacker Front-End (BFE) equipment and Blacker Emergency Mode operation.

Blacker Emergency Mode allows your BFE device and your router to function in emergency situations. When the router is configured to participate in emergency mode and the BFE device is in emergency mode, the router sends address translation information to the BFE device to assist it in sending information.



Our implementation of Blacker Emergency Mode adheres to the specifications outlined in the DCA Blacker Interface Control document, published March 21, 1989.

Your BFE device is configured to be in one of three possible modes as follows:

- Enters emergency mode when requested to by the network. If the router is configured to respond to a BFE device in emergency mode, or if the EXEC command **bfe enter** is used, the router sends address translation information to the BFE device.
- Never enters emergency mode.
- Notifies the router that the emergency mode window is open and waits for the router to tell it to enter emergency mode. If the router is configured to respond to a BFE in emergency mode, or if the EXEC command **bfe enter** is used, the router sends a special address translation packet to the BFE device. The “special” data includes a command to the BFE to enter emergency mode.

Perform these tasks to configure Blacker Emergency Mode:

- Set BFE encapsulation on the router attached to a BFE device.
- Provide address translation information to the BFE device.
- Define the circumstances under which the router will participate in emergency mode.
- Enter Blacker Emergency Mode using the **bfe** EXEC command.

The following tables describe these tasks.

BFE encapsulation operates to map between Class A IP addresses and the X.121 addresses expected by the BFE encryption device. To set BFE encapsulation, perform the following task in interface configuration mode:

Task	Command
Set BFE encapsulation on the router attached to a BFE device.	<b>encapsulation x25 bfe</b>

You must set up a table that provides the address translation information the router sends to the BFE when the BFE is in emergency mode. To do so, perform the following task in interface configuration mode:

Task	Command
Set up the table that lists the BFE nodes (host or gateways) to which the router will send packets.	<b>x25 remote-red</b> <i>host-ip-address</i> <b>remote-black</b> <i>blacker-ip-address</i>

You can define the circumstances under which the router participates in emergency mode and how it will participate in emergency mode. To do so, perform the following tasks in interface configuration mode:

Task	Command
Define the circumstances under which the router will participate in emergency mode.	<b>x25 bfe-emergency</b> {never   always   decision}
Define how a router configured as <b>x25 bfe-emergency decision</b> will participate in emergency mode.	<b>x25 bfe-decision</b> {no   yes   ask}

To set the router to participate in emergency mode or to end participation in emergency mode when your system is so configured, perform the following task in EXEC mode:

Task	Command
Set router to participate in emergency mode.	<b>bfe</b> {enter   leave} <i>type number</i>

For an example of configuring Blacker Emergency mode, see the section “Setting Blacker Emergency Mode Example” at the end of this chapter.

## Configure X.25 Level 3 Optional Parameters and Special Features

The router software allows you to configure the standard Level 2 and Level 3 X.25 parameters and user facilities.

---

**Note** If you connect a router to an X.25 network, use the parameters set by the network administrator. Also, note that the X.25 Level 2 parameters described earlier in this chapter affect X.25 Level 3 operations.

---

This section describes the X.25 parameters, user facilities, and special features you can configure. Which tasks you perform depends upon the structure of your network and the requirements of the service provider. These parameters must be adjusted to match the values used by the X.25 network. It is common for networks to require values different from these defaults.

To configure the optional parameters, user facilities, and special features, perform one or more of the tasks described in the following sections:

- Configure Virtual Circuit Ranges
- Configure the Encapsulation Virtual Circuit Idle Time
- Increase the Number of Virtual Circuits Allowed
- Configure the Ignore Destination Time
- Establish the Packet Acknowledgment Policy
- Configure the X.25 Level 3 Retransmission Timers
- Set X.25 TCP Header Compression
- Define X.25 Host Names

- Define the Default Protocol
- Configure X.25 User Facilities
- Set the Packet Number Modulo
- Suppress or Update the Calling Address
- Suppress the Called Address
- Disable Packet-Level Restarts
- Define the Virtual Circuit Packet Hold Queue Size
- Configure an Interface Alias
- Configure X.25 Bridging

## Configure Virtual Circuit Ranges

The X.25 protocol maintains multiple connections over one physical link between a DTE and a DCE. These connections are called virtual circuits or logical channels (LCs). X.25 can maintain up to 4095 virtual circuits numbered 1 through 4095. An individual virtual circuit is identified by giving its logical channel identifier (LCI) or virtual circuit number (VCN). Many documents use the terms virtual circuit and LC, and VCN, LCN, and LCI interchangeably. Each of these terms refer to the virtual circuit number.

An important part of X.25 operation is the range of virtual circuit numbers. Virtual circuit numbers are broken into four ranges (listed here in numerically increasing order):

- 1 Permanent virtual circuits (PVCs)
- 2 Incoming-only circuits
- 3 Two-way circuits
- 4 Outgoing-only circuits

The incoming-only, two-way, and outgoing-only ranges define the virtual circuit numbers over which a switched virtual circuit (SVC) can be established by placing an X.25 call, much like a telephone network establishes a switched voice circuit when a call is placed.

The rules about DCE and DTE devices initiating calls are as follows:

- Only the DCE device can initiate a call in the incoming-only range.
- Only the DTE device can initiate a call in the outgoing-only range.
- Both the DCE device and the DTE device can initiate a call in the two-way range.

There is no difference in the operation of the SVCs except the restrictions on which a device can initiate a call. These ranges can be used to prevent one side from monopolizing the virtual circuits, which can be useful for X.25 interfaces with a small total number of SVCs available.

Six X.25 parameters define the upper and lower limit of each of the three SVC ranges. A PVC must be assigned a number less than the numbers assigned to the SVC ranges. An SVC range is not allowed to overlap another range.

Table 11-7 lists the virtual circuit types, their ranges, and defaults. Note that the values for these parameters must be the same on both ends of an X.25 link. For connection to a public data network (PDN), these values must be set to the values assigned by the network. An SVC range is unused if its lower and upper limits are set to 0; other than this use for marking unused ranges, virtual circuit 0 is not available.

**Note** Because the X.25 protocol requires the DTE and DCE to have identical virtual circuit ranges, if the interface is up, changes to the virtual circuit range limits will be held until the X.25 protocol RESTARTs the packet service.

---

**Table 11-7 X.25 Virtual Circuits**

Task (Parameter)	Range	Default
Set the LIC.	1–4095	0
Set the HIC.	1–4095	0
Set the LTC.	1–4095	1
Set the HTC.	1–4095	1024 (serial) 4095 (cmns)
Set the LOC.	1–4095	0
Set the HOC.	1–4095	0

For an example of configuring virtual circuit ranges, see the section “Setting Virtual Circuit Ranges Example” later in this chapter.

### Configure the Encapsulation Virtual Circuit Idle Time

The router can clear a datagram transport SVC after a set period of inactivity. Routed SVCs are not timed for inactivity. You can set this time by performing the following task in interface configuration mode:

Task	Command
Set the idle time before an SVC is cleared.	<b>x25 idle</b> <i>minutes</i>

For an example of configuring the SVC idle timer, see the section “Typical X.25 Configuration Example” later in this chapter. See the section “Monitor and Maintain LAPB and X.25” later in this chapter for additional commands that clear virtual circuits.

### Increase the Number of Virtual Circuits Allowed

For X.25 datagram transport, you can establish up to eight switched virtual circuits to one host for each protocol. To increase the number of virtual circuits allowed, perform one or both of the following tasks in interface configuration mode:

Task	Command
Specify the maximum number of switched virtual circuits that can be open simultaneously to one host for each protocol.	<b>x25 nvc</b> <i>count</i>
Set up the LAN protocol-to-X.121 address mapping.	<b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9] x.121-address nvc count</i>

For an example of increasing the number of virtual circuits allowed, see the sections “Typical X.25 Configuration Example” and “DDN X.25 Configuration Example” later in this chapter.

## Configure the Ignore Destination Time

Upon receiving a Clear for an outstanding datagram transport Call Request, the X.25 support code immediately tries another Call Request if it has more traffic to send. This action can overrun some X.25 switches. To define the number of minutes it takes to prevent calls from going to a previously failed destination by, perform the following task in interface configuration mode (incoming calls will still be accepted):

Task	Command
Configure the ignore destination time.	<b>x25 hold-vc-timer</b> <i>minutes</i>

## Establish the Packet Acknowledgment Policy

You can instruct the router to send an acknowledgment packet when it has received a threshold of data packets it has not acknowledged, instead of waiting until its input window is full. A value of 1 will send an acknowledgment for each data packet received if it cannot be acknowledged in an outgoing data packet. This approach improves line responsiveness at the expense of bandwidth. A value of 0 restores the default behavior of waiting until the input window is full.

To establish the acknowledgment threshold, perform the following task in interface configuration mode:

Task	Command
Establish the threshold at which to acknowledge data packets.	<b>x25 th</b> <i>delay-count</i>

## Configure the X.25 Level 3 Retransmission Timers

The X.25 Level 3 retransmission timers determine how long the router must wait before retransmitting various packets. You can set these timers independently.

To set the retransmission timers, perform any of the following tasks in interface configuration mode:

Task	Command
Set DTE T20 Restart Request.	<b>x25 t20</b> <i>seconds</i>
Set DCE T10 Restart Indication.	<b>x25 t10</b> <i>seconds</i>
Set DTE T21 Call Request.	<b>x25 t21</b> <i>seconds</i>
Set DCE T11 Incoming Call.	<b>x25 t11</b> <i>seconds</i>
Set DTE T22 Reset Request.	<b>x25 t22</b> <i>seconds</i>
Set DCE T12 Reset Indication.	<b>x25 t12</b> <i>seconds</i>
Set DTE T23 Clear Request.	<b>x25 t23</b> <i>seconds</i>
Set DCE T13 Clear Indication.	<b>x25 t13</b> <i>seconds</i>

For an example of setting the retransmission timers, see the section “DDN X.25 Configuration Example” later in this chapter.

## Set X.25 TCP Header Compression

We support RFC 1144 TCP/IP header compression on serial lines using HDLC and X.25 encapsulation. The implementation of compressed TCP over X.25 uses one virtual circuit to pass the compressed packets. Any IP traffic (including standard TCP) is carried over separate IP encapsulation virtual circuits.

To set X.25 TCP header compression, perform the following tasks in interface configuration mode:

Task	Command
Implement header compression on IP packets.	<b>ip tcp header-compression</b> [passive]
Allow a separate virtual circuit for compressed packets.	<b>x25 map compressedtcp</b> <i>ip-address x.121-address</i> [option]

## Define X.25 Host Names

To define symbolic host names, perform the following task in global configuration mode:

Task	Command
Define a symbolic host name.	<b>x25 host name</b> <i>x.121-address</i> [ <b> cud</b> <i>call-user-data</i> ]

## Define the Default Protocol

To identify calls with unknown or missing Call User Data as either IP or PAD calls, perform the following task in interface configuration mode:

Task	Command
Interpret incoming calls with unknown CUD.	<b>x25 default</b> <i>protocol</i>

## Configure X.25 User Facilities

The X.25 software provides commands to support X.25 user facilities—options specified by the creators of the X.25 Recommendation—that allow you to implement features such as accounting, user identification, and flow control negotiation. The facilities configured by the **x25 map** commands are on a per-peer basis; the **x25 facility** commands specify the values sent for calls originated by the interface. Routed calls are not affected by the facilities specified for the outgoing interface.

To set the supported X.25 user facilities, perform one or more of the following tasks in interface configuration mode:

<b>Task (User Facility)</b>	<b>Command</b>
Select closed user group.	<b>x25 facility cug</b> <i>number</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address cug number</i>
Set flow control parameter negotiation values to request on outgoing calls.	<b>x25 facility packetsize</b> <i>in-size out-size</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address packetsize in-size out-size</i> <b>x25 facility windowsize</b> <i>in-size out-size</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address windowsize in-size out-size</i>
Set reverse charging.	<b>x25 facility reverse</b> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address reverse</i>
Allow reverse charging acceptance.	<b>x25 accept-reverse</b> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address accept-reverse</i>
Select throughput class negotiation.	<b>x25 facility throughput</b> <i>in out</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address throughput in out</i>
Select transit delay.	<b>x25 facility transit-delay</b> <i>number</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address transit-delay number</i>
Set the Recognized Private Operation Agency (RPOA) to use.	<b>x25 facility rpoa</b> <i>name</i> <b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9] x.121-address rpoa name</i>
Set the Cisco standard network user identification.	<b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9] x.121-address nuid username password</i>
Set a user-defined network user identification allowing format determined by the network administrator	<b>x25 map</b> <i>protocol address [protocol2 address2...protocol9 address9]</i> <i>x.121-address nudata string</i>

Additionally, the D-bit is supported and passed through transparently. Both restricted and unrestricted fast select are also supported and are transparently handled by the software. No configuration is required for use of the D-bit or fast select facilities.

### Set the Packet Number Modulo

Our implementation of X.25 supports the extended packet sequence numbering. To use the extended packet sequence, perform the following task in interface configuration mode:

Task	Command
Set the packet numbering modulo.	<b>x25 modulo <i>modulus</i></b>

**Note** Because the X.25 protocol requires the DTE and DCE to have identical modulus, if the interface is up, changes to the modulo will be held until the X.25 protocol restarts the packet service.

### Suppress or Update the Calling Address

You can omit or update the calling (source) address in outgoing calls. Either of these options might be required by networks that expect only subaddresses or that expect the interface’s address in the calling address field. To suppress or update the calling address, perform the appropriate task in interface configuration mode:

Task	Command
Omit the calling (source) X.121 address in outgoing calls.	<b>x25 suppress-calling-address</b>
Replace the calling (source) X.121 address in switched calls.	<b>x25 use-source-address</b>

### Suppress the Called Address

You can omit the called (destination) address in outgoing calls. This option is required for networks that expect only subaddresses in the called address field. To suppress the called address, perform the following task in interface configuration mode:

Task	Command
Omit the called (destination) X.121 address in outgoing calls	<b>x25 suppress-called-address</b>

### Disable Packet-Level Restarts

By default, a packet-level restart is performed when the link level is reset. This behavior can be disabled for networks that do not allow it, but disabling this behavior can cause anomalous packet layer behavior. To disable packet-level restarts, perform the following task in interface configuration mode:

Task	Command
Disable packet-level restarts.	<b>no x25 linkrestart</b>



## Define the Virtual Circuit Packet Hold Queue Size

To define the maximum number of packets that can be held while a virtual circuit is unable to send data, perform the following task in interface configuration mode:

Task	Command
Define the virtual circuit packet hold queue size.	<b>x25 hold-queue</b> <i>queue-size</i>

## Configure an Interface Alias

You can supply alias X.121 addresses for an interface. This allows the interface to accept data encapsulation calls to an address other than the address defined by the **x25 address** command.

Local processing (for example, IP encapsulation) can be performed only for incoming calls whose destination X.121 address matches the serial interface or alias of the interface.

To configure an alias, perform the following task in interface configuration mode:

Task	Command
Supply an alias for other X.121 addresses that will be accepted on the interface.	<b>x25 route</b> [ <i>#position</i> ] <i>x121-address</i> [ <i>cu</i> <i>pattern</i> ] <b>alias</b> <i>type number</i>

## Configure X.25 Bridging

Our transparent bridging software supports bridging over X.25 virtual circuits. To enable the X.25 bridging capability, perform the following task in interface configuration mode:

Task	Command
Define bridging of X.25 frames.	<b>x25 map bridge</b> <i>x.121-address</i> <b>broadcast</b> [ <i>option</i> ]

## Monitor and Maintain LAPB and X.25

To monitor and maintain X.25 and LAPB, perform any of the following tasks in EXEC mode:

Task	Command
Clear all virtual circuits at once (calls originated, terminated, and switched by the router are cleared), or clear the single virtual circuit specified.	<b>clear x25-vc</b> <i>type number</i> [ <i>lcn</i> ]
Display CMNS information.	<b>show cmns</b> [ <i>type number</i> ]
Display operation statistics for an interface.	<b>show interfaces serial</b> <i>number</i>
Display CMNS connections over LLC2.	<b>show llc2</b>
Display the protocol-to-X.121 address map.	<b>show x25 map</b>
Display the one-to-one mapping of the host IP addresses and the remote BFE device's IP addresses.	<b>show x25 remote-red</b>
Display routes assigned by the <b>x25 route</b> command.	<b>show x25 route</b>
Display details of active virtual circuits.	<b>show x25 vc</b> [ <i>lcn</i> ]

---

**Note** See the “X.25 Cause and Diagnostic Codes” appendix in the *Debug Command Reference* publication for a description of X.25 diagnostic codes that can appear in these **show** command displays.

---

The following sections provide examples to help you understand how to configure LAPB and X.25 for your network:

- Typical LAPB Configuration Example
- Typical X.25 Configuration Example
- Setting Virtual Circuit Ranges Example
- Switching a PVC on the Same Router Example
- X.25 Route Address Pattern Matching Example
- X.25 Routing Example
- Using a PVC to Exchange IP Traffic Example
- Establishing a Connection between Two PVCs Example
- Creating and Configuring a Point-to-Point Subinterface Example
- Enabling Remote Tunneling Example
- Enabling CMNS for X.121 and MAC Addresses Example
- Switching CMNS over a PDN Example
- Switching CMNS over Leased Lines Example
- DDN X.25 Configuration Example
- Setting Blacker Emergency Mode Example
- Configuring X.25 to Allow Ping Support over Multiple Lines Example
- Netbooting over X.25 Example

### Typical LAPB Configuration Example

In the following example, the frame size (N1), window size (k), and maximum retransmission (N2) parameters retain their default values. The **encapsulation** interface configuration command sets DCE operation to carry a single protocol, IP by default, and the **lapb t1** interface configuration command sets the retransmission timer to 4,000 milliseconds (4 seconds) for a link with a long delay or slow connecting DTE device.

```
interface serial 3
encapsulation lapb dce
lapb t1 4000
```

### Typical X.25 Configuration Example

The following example shows the complete configuration for a serial interface connected to a commercial X.25 PDN for routing the IP protocol. The IP subnetwork address 131.108.9.0 has been assigned for the X.25 network.

---

**Note** When routing IP over X.25, the X.25 network must be treated as a single IP network or subnetwork. Map entries for routers with addresses on subnetworks other than the one on which the interface's IP address is stored are ignored by the routing software. Additionally, all routers using the subnet number should have map entries for all others. There are also issues with the broadcast flag, which apply both to IP and to other protocols with dynamic routing.

---

```

interface serial 2
ip address 131.108.9.1 255.255.255.0
!
encapsulation X25
!
! The "bandwidth" command is not part of the X.25
! configuration; it is especially important to understand that it does not
! have any connection with the X.25 entity of the same name.
! "bandwidth" commands are used by IP routing processes (currently only IGRP)
! to determine which lines are the best choices for traffic.
! Since the default is 1544 Kbaud, and X.25 service at that rate is not generally
! available, most X.25 interfaces that are being used with IGRP in a
! real environment will have "bandwidth" settings.
!
! This is a 9.6 Kbaud line:
!
bandwidth 10
!
! These Level 3 parameters are default flow control values; they need to
! match the PDN defaults. The values used by an SVC are negotiable on a per-call basis:
!
x25 win 7
x25 wout 7
x25 ips 512
x25 ops 512
!
! You must specify an X.121 address to be assigned to the X.25
! interface by the PDN.
!
x25 address 31370054065
!
! The following Level 3 parameters have been set to match the network.
! You generally need to change some Level 3 parameters, most often
! those listed below. You might not need to change any Level 2
! parameters, however.
!
x25 htc 32
x25 idle 5
x25 nvc 2
!
! The following commands configure the X.25 map. If you want to exchange
! routing updates with any of the routers, they would need
! "broadcast" flags.
! If the X.25 network is the only path to the routers, static routes are
! generally used to save on packet charges. If there is a redundant
! path, it might be desirable to run a dynamic routing protocol.
!
x25 map IP 131.108.9.3 31370019134 ACCEPT-REVERSE
! ACCEPT-REVERSE allows collect calls
x25 map IP 131.108.9.2 31370053087
!
! If the PDN cannot handle fast back-to-back frames, use the
! "transmitter-delay" command to slow down the interface.
!

```

```
transmitter-delay 1000
```

### Setting Virtual Circuit Ranges Example

The following example sets the following virtual circuit ranges: 5 to 20 dedicated to incoming calls only (from the DCE to the DTE), 25 to 1024 for either incoming or outgoing calls, no virtual circuit range dedicated to outgoing calls (from the DTE to the DCE). Up to four permanent virtual circuits can be defined on virtual circuits 1 through 4.

```
x25 lic 5
x25 hic 20
x25 ltc 25
```

### Switching a PVC on the Same Router Example

In the following example, there is a PVC connected between two serial interfaces on the same router. In this type of interconnection configuration, the destination interface must be specified along with the PVC number on that interface. To make a working PVC connection, two commands must be specified, each pointing to the other.

```
interface serial 0
encapsulation x25
x25 ltc 5
x25 pvc 1 interface serial 1 pvc 4
!
interface serial 1
encapsulation x25
x25 ltc 5
x25 pvc 4 interface serial 0 pvc 1
```

### X.25 Route Address Pattern Matching Example

The following example shows how to indicate that X.25 calls to addresses whose first four Data Network Identification Code (DNIC) digits are 1111 should be routed to interface serial 3, but that the DNIC field in the addresses presented to the equipment connected to that interface should be changed to 2222. The \1 in the rewrite pattern indicates the portion of the original address matched by the digits following the 1111 DNIC.

```
x25 route ^1111(.*) substitute-dest 2222\1 interface serial 3
```

The following example shows a more contrived command intended to illustrate the power of the rewriting scheme:

```
x25 route ^(..)...(..)...(..)(..)$ substitute-dest \2\4\3\1 interface serial 0
```

S1046a

It causes all X.25 calls with 14-digit called addresses to be routed through interface serial 0. The incoming DNIC field would be moved to the end of the address. The fifth, sixth, ninth, and tenth digits would be deleted, and the thirteenth and fourteenth would be moved before the eleventh and twelfth.

## X.25 Routing Example

The following examples illustrate how to enable an X.25 switch, how to enable call forwarding, and how to configure a router on a Tymnet/PAD switch to accept and forward calls.

This first example shows how to enable X.25 switching, as well as how to enter routes into the X.25 routing table:

```
! Enable X.25 forwarding
x25 routing
!
! Enter routes into the table. Without a positional parameter, entries
! are appended to the end of the table
x25 route ^100$ interface serial 0
x25 route 100 cud ^pad$ interface serial 2
x25 route 100 interface serial 1
x25 route ^3306 interface serial 3
x25 route .* ip 10.2.0.2
```

The routing table forwards calls for X.121 address 100 out interface serial 0. Otherwise, if the X.121 address contains 100 anywhere within it and contains no Call User Data, or the Call User Data is not the string “pad”, it is forwarded onto serial 1. If the X.121 address contains the digits 100 and the Call User Data is the string “pad”, the call is forwarded onto serial 2. All X.121 addresses that do not match the first three routes are checked for a DNIC of 3306 as the first four digits. If they do match, they are forwarded over serial 3. All other X.121 addresses will match the fifth entry, which is a match-all pattern and will have a TCP connection established to the IP address 10.2.0.2. The router at 10.2.0.2 will then route the call according to its X.25 routing table.

This second example configures a router that sits on a Tymnet PAD/switch to accept calls and have them forwarded to a DEC VAX system. This feature permits running X.25 network over a generalized, existing IP network, thereby making it unnecessary to get another physical line for one protocol. The router positioned next to the DEC VAX system is configured with X.25 routes, as follows:

```
x25 route vax-x121-address interface serial 0
x25 route .* ip cisco-on-tymnet-ipaddress
```

This routes all calls to the DEC VAX X.121 address out to serial 0, where the VAX is connected running PSI. All other X.121 addresses are forwarded to the *cisco-on-tymnet* address using its IP address. This takes all outgoing calls from the VAX and sends them to *cisco-on-tymnet* for further processing.

On the router named *cisco-on-tymnet*, you would enter these commands:

```
x25 route vax-x121-address ip cisco-on-vax
x25 route .* interface serial 0
```

This forces all calls with the VAX X.121 address to be sent to the router with the VAX connected to it. All other calls with X.121 addresses are forwarded out to Tymnet. If Tymnet can route them, a Call Accepted packet is returned, and everything proceeds normally. If Tymnet cannot handle it, it clears the call, and the Clear Request packet is forwarded back toward the VAX.

## Using a PVC to Exchange IP Traffic Example

The following example, illustrated in Figure 11-4, demonstrates how to use the PVC to exchange IP traffic between Router X and Router Y.

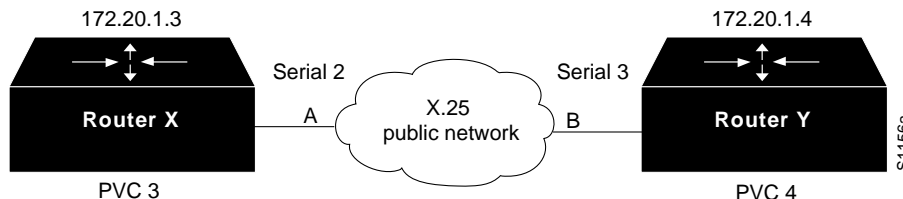


Figure 11-4 Establishing an IP Encapsulation PVC through an X.25 Network

### Configuration for Router X

```
interface serial 2
ip address 131.108.1.3 255.255.255.0
x25 map ip 131.108.1.4 0
x25 pvc 4 ip 131.108.1.4
```

### Configuration for Router Y

```
interface serial 3
ip address 131.108.1.4 255.255.255.0
x25 map ip 131.108.1.3 0
x25 pvc 4 ip 131.108.1.3
```

In this example, the PDN has established a PVC through its network connecting PVC number 3 of access point A to PVC number 4 of access point B. On Router X, a connection is established between Router X and Router Y's IP address, 131.108.1.4. On Router Y, a connection is established between Router Y and Router X's IP address, 131.108.1.3.

## Establishing a Connection between Two PVCs Example

In the following example, a connection is established between two PVCs across a LAN. Because the connection is remote (across the LAN), the tunneling command is used. This example establishes a PVC between Router X, Serial 0, PVC1 and Router Y, Serial 1, PVC 2. Figure 11-5 provides a visual representation of the configuration.

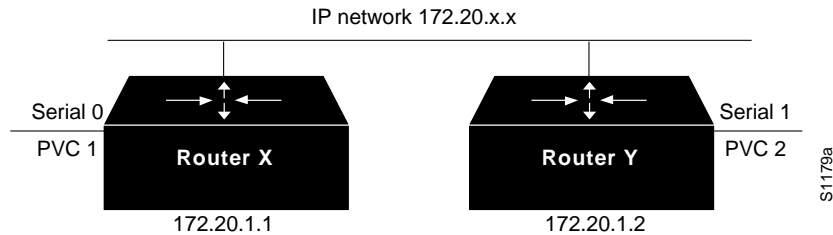


Figure 11-5 X.25 Tunneling Connection

### Configuration for Router X

```
interface serial 0
x25 pvc 1 tunnel 131.108.1.2 interface serial 1 pvc 2
```

### Configuration for Router Y

```
interface serial 1
x25 pvc 2 tunnel 131.108.1.1 interface serial 0 pvc 1
```

## Creating and Configuring a Point-to-Point Subinterface Example

The following example creates a point-to-point subinterface, maps IP and AppleTalk to a remote host, and creates an encapsulating PVC for DECnet to the same remote host, identified by the X.121 address in the commands.

```
interface Serial0.1 point-to-point
x25 map ip 131.108.170.90 170090 broadcast
x25 map appletalk 131.108.170.90 170090 broadcast
x25 pvc 1 decnet 1.2 170090 broadcast
```

## Enabling Remote Tunneling Example

In Figure 11-6, the connection between points A and B is switched, and the connections between points A or B and C are tunneled.

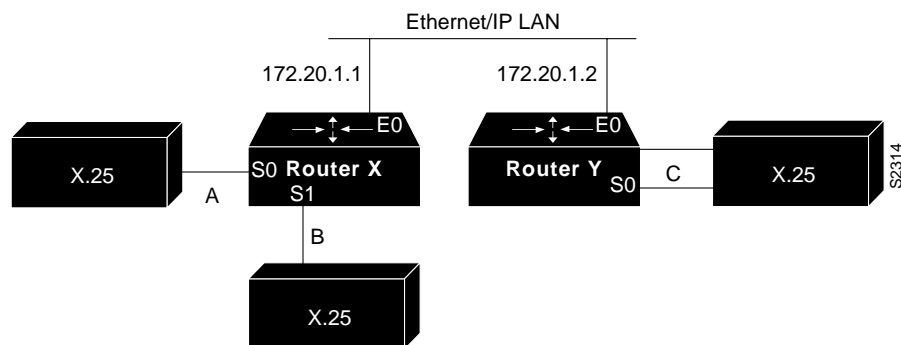


Figure 11-6 Local Switching and Remote Tunneling PVCs

### Configuration for Router X

```
interface ethernet 0
ip address 131.108.1.1 255.255.255.0
!
interface serial 0
x25 ltc 5
x25 pvc 1 interface serial 1 pvc 1
x25 pvc 2 tunnel 131.108.1.2 interface serial 0 pvc 1
!
interface serial 1
x25 ltc 5
x25 pvc 1 interface serial 0 pvc 1
x25 pvc 2 tunnel 131.108.1.2 interface serial 0 pvc 2
```

### Configuration for Router Y

```
interface ethernet 0
ip address 131.108.1.2 255.255.255.0
!
interface serial 0
x25 ltc 5
x25 pvc 1 tunnel 131.108.1.1 interface serial 0 pvc 2
x25 pvc 2 tunnel 131.108.1.1 interface serial 1 pvc 2
```

## Enabling CMNS for X.121 and MAC Addresses Example

The following examples illustrate enabling CMNS and configuring X.121 and MAC address mappings:

```
interface ethernet 0
cmns enable
x25 map cmns 38.8261.1000.0150.1000.17 0000.0c00.ff89
! Above maps NSAP to MAC-address on Ethernet0
!
interface serial 0
encapsulation x25
x25 map cmns 38.8261.1000.0150.1000.18 3110451
! Above maps NSAP to X.121-address on Serial0
! assuming the link is over a PDN
!
interface serial 1
encapsulation x25
x25 map cmns 38.8261.1000.0150.1000.20
! Above specifies cmns support for Serial1
! assuming that the link is over a leased line
```

## Switching CMNS over a PDN Example

The following example depicts switching CMNS over a packet-switched public data network (PDN). Figure 11-7 illustrates the general network topology for a CMNS switching application where calls are being made between resources on opposite sides of a remote link to Host A (on an Ethernet) and Host B (on a Token Ring), with a PDN providing the connection.



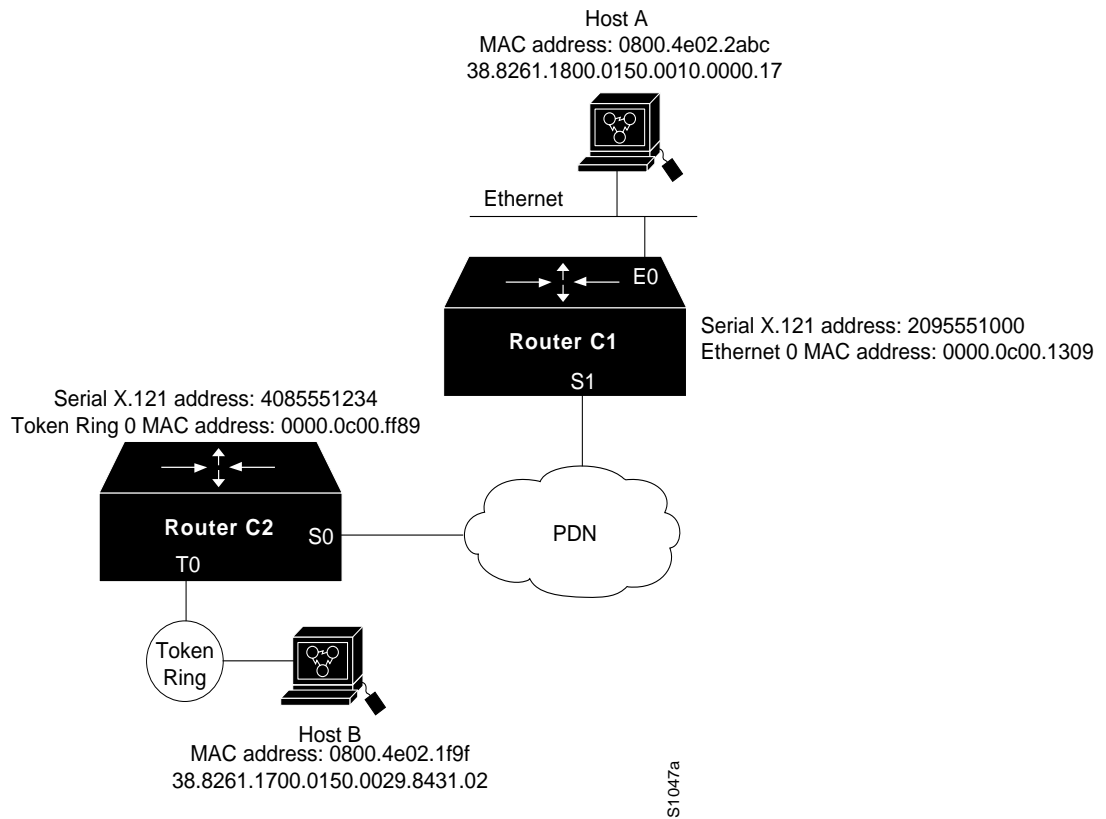


Figure 11-7 Example Network Topology for Switching CMNS over a PDN

The following configuration listing allows resources on either side of the PDN to call Host A or Host B. This configuration allows traffic intended for the remote NSAP address specified in the **x25 map cmns** commands (for the serial ports) to be switched through the serial interface for which CMNS is configured.

### Configuration for Router C2

```

! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP prefix 38.8261.17
! will be switched to MAC address 0800.4e02.1f9f
! through Token Ring 0
!
interface token 0
cmns enable
x25 map cmns 38.8261.17 0800.4e02.1f9f
!
! This configuration specifies that traffic from any other interface
! on Cisco Router C2 that is intended for any NSAP address with
! NSAP-prefix 38.8261.18 will be switched to
! X.121 address 2095551000 through Serial 0
!
interface serial 0
encapsulation x25
x25 address 4085551234
x25 map cmns 38.8261.18 2095551000

```

### Configuration for Router C1

```

! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.18
! will be switched to MAC address 0800.4e02.2abc through Ethernet 0
!
interface ethernet 0
cmns enable
x25 map cmns 38.8261.18 0800.4e02.2abc
!
! This configuration specifies that traffic from any other interface
! on Cisco Router C1 that is intended for any NSAP address with
! NSAP-prefix 38.8261.17 will be switched to X.121 address
! 4085551234 through Serial 1
!
interface serial 1
encapsulation x25
x25 address 2095551000
x25 map cmns 38.8261.17 4085551234
    
```

### Switching CMNS over Leased Lines Example

The following example illustrates switching CMNS over a leased line. Figure 11-8 illustrates the general network topology for a CMNS switching application where calls are being made by resources on the opposite sides of a remote link to Host C (on an Ethernet) and Host D (on a Token Ring), with a dedicated leased line providing the connection.

The following configuration listing allows resources on either side of the leased line to call Host C or Host D. This configuration allows traffic intended for the remote NSAP address specified in the **x25 map cmns** commands (for the serial ports) to be switched through the serial interface for which CMNS is configured.

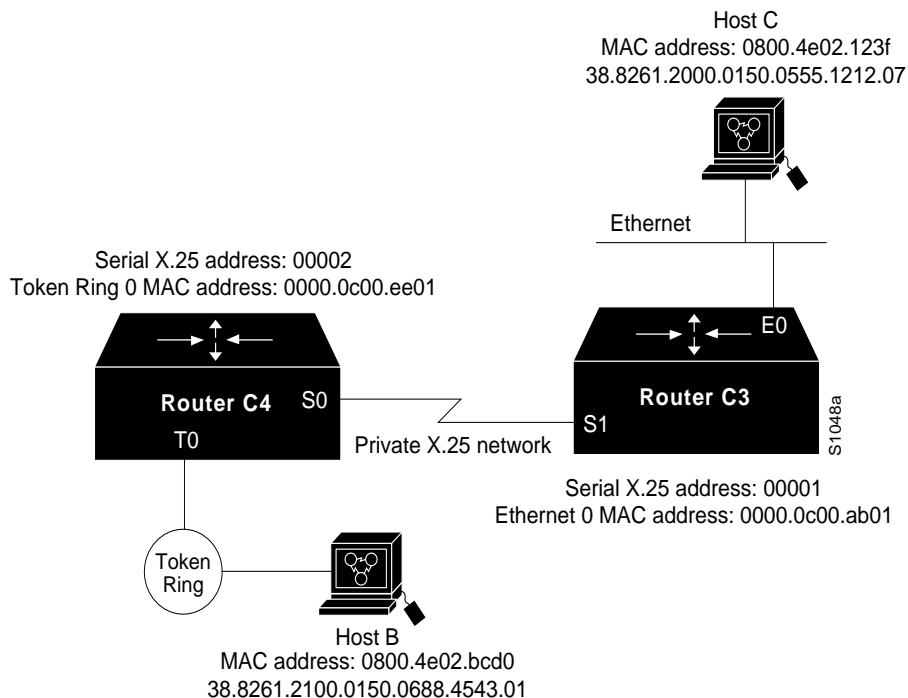


Figure 11-8 Example Network Topology for Switching CMNS over a Leased Line

A key difference for this configuration compared with the previous example is that with no PDN, the specification of an X.121 address in the **x25 map cmns** command is not necessary. The specification of an X.25 address also is not needed, but is included for symmetry with the previous example.

### Configuration for Router C4

```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.21
! will be switched to MAC address 0800.4e02.bcd0 through Token Ring 0
!
interface token 0
cmns enable
x25 map cmns 38.8261.21 0800.4e02.bcd0
!
! This configuration specifies that traffic from any other interface
! on Cisco Router C4 that is intended for any NSAP address with
! NSAP-prefix 38.8261.20 will be switched through Serial 0
!
interface serial 0
encapsulation x25
x25 address 00002
x25 map cmns 38.8261.20
```

### Configuration for Router C3

```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.20
; will be switched to MAC address 0800.4e02.123f through Ethernet 0
!
interface ethernet 0
cmns enable
x25 map cmns 38.8261.20 0800.4e02.123f
!
! This configuration specifies that traffic from any other interface
! on Cisco Router C3 that is intended for any NSAP address with
! NSAP-prefix 38.8261.21 will be switched through Serial 1
!
interface serial 1
encapsulation x25
x25 address 00001
x25 map cmns 38.8261.21
```

## DDN X.25 Configuration Example

The following example illustrates how to configure a router interface to run DDN X.25:

```
interface serial 0
ip address 192.31.7.50 255.255.255.240
encapsulation x25 ddn
x25 win 6
x25 wout 6
x25 ips 1024
x25 ops 1024
x25 t20 10
x25 t21 10
x25 t22 10
x25 t23 10
x25 nvc 2
x25 map IP 192.31.7.49 000000010300 BROADCAST
```

### Setting Blacker Emergency Mode Example

In the following example, interface serial 0 is configured to require an EXEC command from the administrator before it participates in emergency mode. The host IP address is 21.0.0.12, and the address of the remote BFE unit is 21.0.0.1. When the BFE enters emergency mode, the router will prompt the administrator for the EXEC command **bfe enter** to direct the router to participate in emergency mode.

```
interface serial 0
ip address 21.0.0.2 255.0.0.0
encapsulation x25 bfe
x25 bfe-emergency decision
x25 remote-red 21.0.0.12 remote-black 21.0.0.1
x25 bfe-decision ask
```

### Configuring X.25 to Allow Ping Support over Multiple Lines Example

For **ping** commands to work in an X.25 environment (when load sharing over multiple serial lines), you must include entries for all adjacent interface IP addresses in the **x25 map** command for each serial interface. The following example illustrates this point.

Consider two routers, Router A and Router B, communicating with each other over two serial lines via an X.25 PDN (see Figure 11-9) or over leased lines. In either case, all serial lines must be configured for the same IP subnet address space. In order to allow for successful **ping** commands, the configuration might be as in the lists that follow. In any event, a similar configuration is required for the same subnet IP addresses to work across X.25.

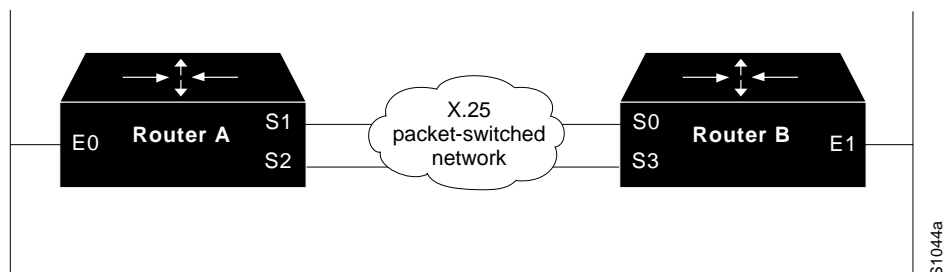


Figure 11-9 Parallel Serial Lines to X.25 Network

---

**Note** All four serial ports configured for the two routers in the following configuration example must be assigned to the same IP subnet address space. In this case, the subnet is 131.108.170.0.

---

**Configuration for Router A**

```

interface serial 1
ip 131.108.170.1 255.255.255.0
x25 address 31370054068
x25 map ip 131.108.170.3 31370054065
x25 map ip 131.108.170.4 31370054065
!
interface serial 2
ip 131.108.170.2 255.255.255.0
x25 address 31370054069
x25 map ip 131.108.170.4 31370054067
x25 map ip 131.108.170.3 31370054067
! allow either destination address
x25 31370054068 alias serial2
x25 31370054069 alias serial1

```

**Configuration for Router B**

```

interface serial 0
ip 131.108.170.3 255.255.255.0
x25 address 31370054065
x25 map ip 131.108.170.1 31370054068
x25 map ip 131.108.170.2 31370054068
!
interface serial 3
ip 131.108.170.4 255.255.255.0
x25 address 31370054067
x25 map ip 131.108.170.2 31370054069
x25 map ip 131.108.170.1 31370054069
! allow either destination address
x25 31370054065 alias serial3
x25 31370054067 alias serial0

```

**Netbooting over X.25 Example**

When netbooting over X.25, you cannot netboot via a broadcast. You must netboot from a specific host. Also, an **x25 map** command must exist for the host that you netboot from. The **x25 map** command is used to map an IP address into an X.121 address. There must be an **x25 map** command that matches the IP address given on the **boot system** command line. The following is an example of such a configuration:

```

boot system gs3-k.100 131.108.126.111
!
interface Serial 1
ip address 131.108.126.200 255.255.255.0
encapsulation X25
x25 address 10004
x25 map IP 131.108.126.111 10002 broadcast
lapb nl 12040
clockrate 56000

```

In this case, 10002 is the X.121 address of the remote router that can get to host 131.108.126.111.

The remote router must have the following **x25 map** entry:

```
x25 map IP 131.108.126.200 10004 broadcast
```

This entry allows the remote router to return a boot image (from the netboot host) to the router netbooting over X.25.

