CHAPTER 2

PPP

Objectives

After completing this chapter, you should be able to answer the following questions:

- What are the fundamental concepts of point-to-point serial communication?
- What are the key concepts of PPP?
- What commands are used to configure PPP encapsulation?
- What commands are used to configure PAP and CHAP authentication?

Key Terms

This chapter uses the following key terms. You can find the definitions in the glossary at the end of the book.

point-to-point connections *page 56*

Point-to-Point Protocol (PPP) *page 56*

Password Authentication Protocol (PAP) *page 56*

Challenge Handshake Authentication Protocol (CHAP) *page 56*

clock skew *page 57*

Data Terminal Ready (DTR) *page 61*

Data Set Ready (DSR) *page 61*

Clear to Send (CTS) *page 61*

time-division multiplexing (TDM) *page 61*

transmission link *page 62*

data streams *page 62*

statistical time-division multiplexing (STDM) *page 63*

buffer *page 63*

SONET *page 64*

DS0 (digital signal level zero) *page 64*

Frame Relay access device *page 66*

null modem *page 68*

Universal Asynchronous Receiver/Transmitter (UART) *page 71*

Internetwork Packet Exchange (IPX) *page 73*

Serial Line Internet Protocol (SLIP) *page 73*

Link Access Procedure, Balanced (LAPB) *page 73*

cell relay *page 73*

E3 *page 73*

bit-oriented *page 73*

Synchronous Data Link Control (SDLC) *page 73*

primary station *page 75*

Cisco 7000 *page 82*

trunk *page 83*

Link Control Protocol (LCP) *page 83*

Network Control Protocols (NCP) *page 83*

Novell IPX *page 83*

SNA Control Protocol *page 83*

fragmentation *page 97*

reassembly *page 97*

Message Digest 5 (MD5) *page 111*

TACACS/TACACS+ *page 116*

This chapter starts your exploration of WAN technologies by introducing point-to-point communications and Point-to-Point Protocol (PPP).

One of the most common types of WAN connections is the point-to-point connection using PPP as the Layer 2 protocol. ***Point-to-point connections*** are used to connect LANs to service provider WANs and to connect LAN segments within an Enterprise network. A LAN-to-WAN point-to-point connection is also called a serial connection or leased-line connection, because the lines are leased from a carrier (usually a telephone company) and are dedicated for use by the company leasing the lines. Companies pay for a continuous connection between two remote sites, and the line is continuously active and available. Understanding how point-to-point communication links function to provide access to a WAN is important to an overall understanding of how WANs function.

Point-to-Point Protocol (PPP) provides multiprotocol LAN-to-WAN connections handling TCP/IP, IPX, and AppleTalk simultaneously. It can be used over twisted pair, fiber-optic lines, and satellite transmission. PPP provides transport over WAN technologies such as ATM, Frame Relay, ISDN, and optical links. These technologies are discussed later.

In modern networks, security is a key concern. PPP allows you to authenticate connections using either ***Password Authentication Protocol (PAP)*** or the more effective ***Challenge Handshake Authentication Protocol (CHAP)***.

In this chapter you will also learn about the key concepts of serial communications and how to configure and troubleshoot a PPP serial connection on a Cisco router.

Introducing Serial Communications

Serial communication is the process of transmitting a single bit at a time over a communications circuit or channel.

How Does Serial Communication Work?

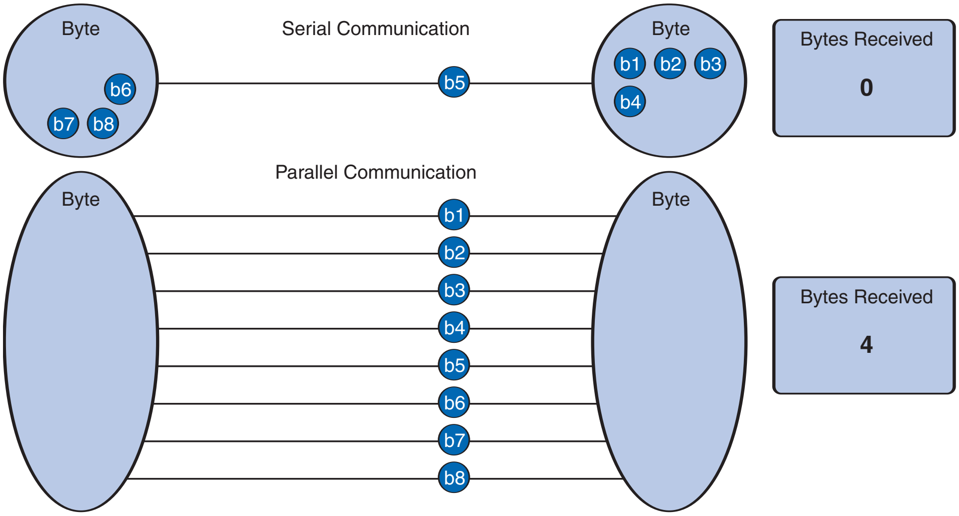
You know that most PCs have both serial and parallel ports. You also know that electricity can move at only one speed. To transfer data quicker, it can be compressed and therefore require fewer bits to be transmitted. An alternative method is to transmit the bits simultaneously, as done in computers with parallel connections. Computers use relatively short parallel connections between interior components, but they use a serial bus to convert signals for most external communications. Let's compare serial and parallel communications.

With a serial connection, information is sent across one wire, one data bit at a time. The nine-pin serial connector on most PCs uses two loops of wire, one in each direction, for data communication, plus additional wires to control the flow of information. In any given direction, data is still flowing over a single wire.

A parallel connection sends the bits over more wires simultaneously. In the case of the 25-pin parallel port on your PC, eight data-carrying wires carry 8 bits simultaneously. Because eight wires carry the data, the parallel link theoretically transfers data eight times faster than a serial connection. So, based on this theory, a parallel connection sends a byte in the same amount of time a serial connection takes to send a bit.

In Figure 2-1, the serial connection sends 1 bit at a time, while the parallel connection sends 8 bits at a time. Notice that the serial connection has sent 4 bits and is currently sending the fifth bit, while the parallel connection has already sent 4 bytes.

Figure 2-1 Serial and Parallel Communication

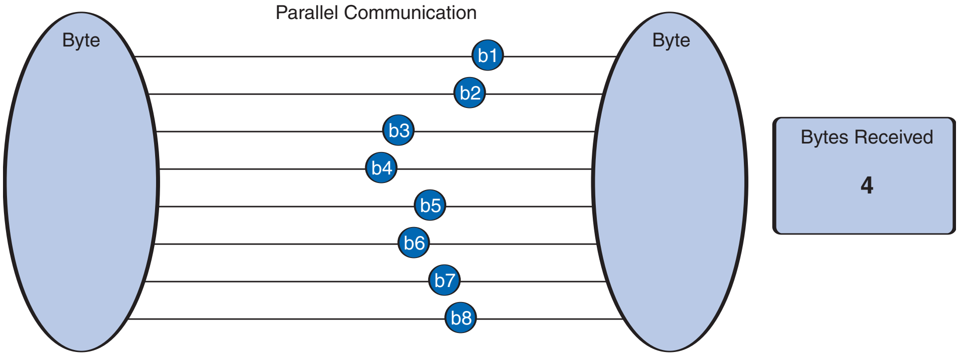


This explanation brings up some questions. What is meant by "theoretically faster"? If parallel is theoretically faster than serial, is parallel better suited for connecting to a WAN? It would initially seem that a serial link must be inferior to a parallel one, because it can transmit less data at each clock tick. However, serial links can be clocked considerably faster than parallel links and therefore can achieve faster data rates. As well, parallel links are limited to shorter distances because of the effects of clock skew and crosstalk interference.

For instance, in a parallel connection, it is wrong to assume that the 8 bits leaving the sender at the same time arrive at the receiver at the same time. Rather, some of the bits get there later than others. This is known as ***clock skew***. Overcoming clock skew is not trivial. The receiving end must synchronize itself with the transmitter and then wait until all the bits have arrived. The process of reading, waiting, latching, waiting for clock signal, and transmitting the 8 bits adds time to the transmission. In parallel communications, a latch is a data storage system used to store information in sequential logic systems. The more wires you use and the farther the connection reaches, this compounds the problem and adds delay. The need for clocking slows parallel transmission well below theoretical expectations. Clock skew is not a factor with serial links, because it has only one channel to transmit on.

In Figure 2-2, notice that the bits of the parallel connection do not arrive at the same time due to clock skew.

Figure 2-2 Clock Skew

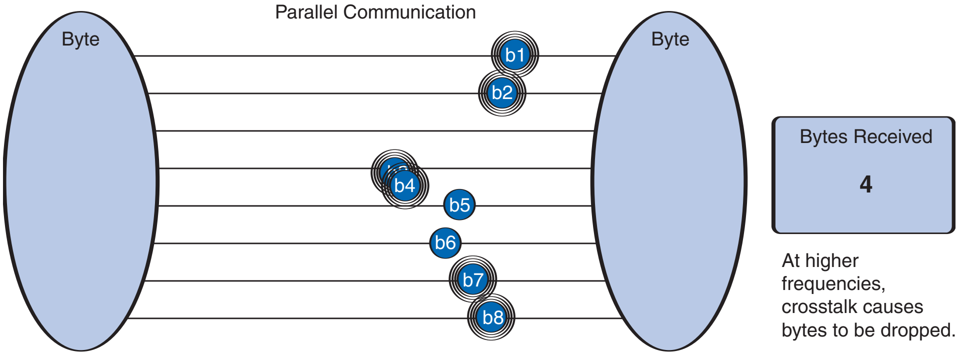


Parallel wires are physically bundled in a parallel cable, and signals can imprint themselves on each other, making the cable more susceptible to crosstalk. The possibility of crosstalk across the wires requires more processing, especially at higher frequencies.

The serial buses on computers and routers compensate for crosstalk before transmitting the bits. Because serial cables have fewer wires, less crosstalk occurs, and network devices transmit serial communications at higher, more efficient frequencies.

In Figure 2-3, bits 1, 2, 7, and 8 are experiencing some crosstalk, but bits 5 and 6 are not. Bits 3 and 4 have encountered too much crosstalk and therefore are dropped.

Figure 2-3 Crosstalk


Parallel Communication
Byte
Byte
b1
b2
b4
b5
b6
b7
b8
Bytes Received
4
At higher frequencies, crosstalk causes bytes to be dropped.

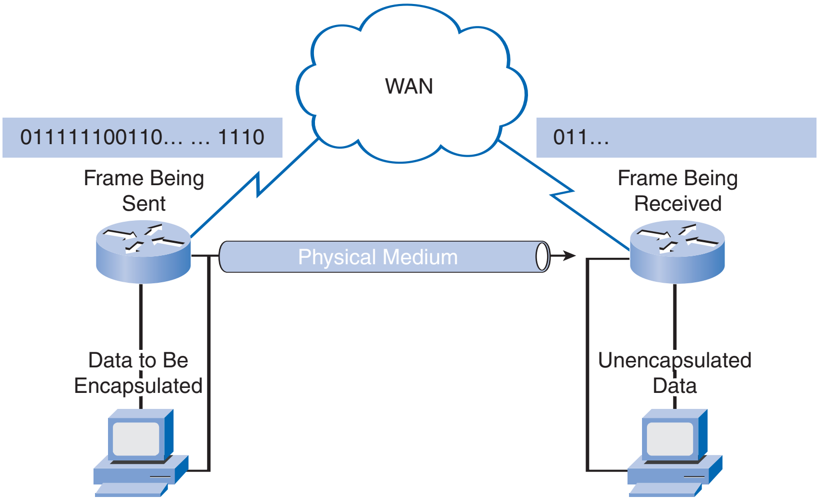
In most cases, serial communications are considerably cheaper to implement. Serial communications use fewer wires, cheaper cables, and fewer connector pins.

Serial Communication Standards

All long-haul communications and most computer networks use serial connections, because the cost of cable and synchronization difficulties makes parallel connections impractical. The most significant advantage is simpler wiring. Also, serial cables can be longer than parallel cables, because much less interaction (crosstalk) occurs among the conductors in the cable. In this chapter, we will confine our consideration of serial communications to those connecting LANs to WANs.

Figure 2-4 is a simple representation of serial communication. The sending router encapsulates the data using PPP. The encapsulated PPP frame is sent on a physical medium to the WAN. There are various ways to traverse the WAN, but the receiving router uses the same communications protocol to unencapsulate the frame when it arrives.

Figure 2-4 Serial Communication



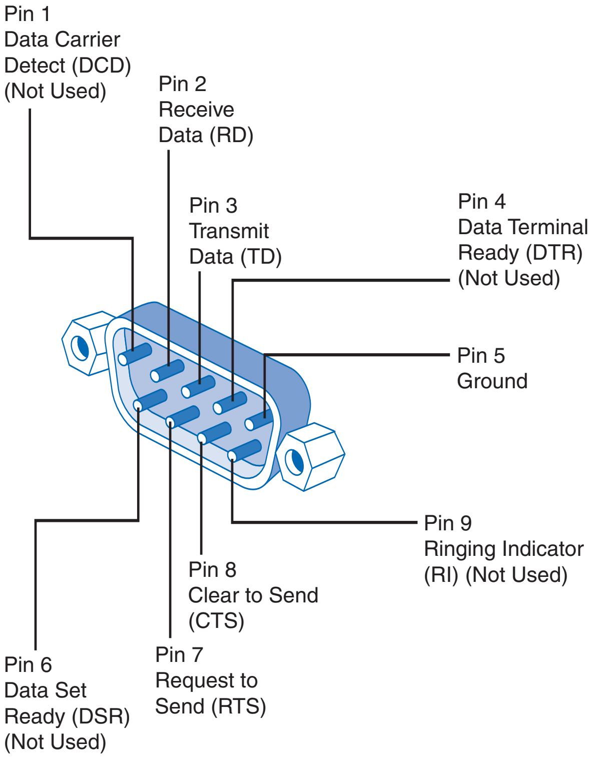
There are many different serial communication standards, each one using a different signaling method. Three key serial communication standards affect LAN-to-WAN connections:

- **RS-232**: Most serial ports on PCs conform to the RS-232C standard or the newer RS-422 and RS-423 standards. Although the standard defines a nine-pin connector and a 25-pin connector, the nine-pin connector is more commonly implemented. A serial port is a general-purpose interface that can be used for almost any type of device, including modems, mice, and printers. Many network devices use RJ-45 connectors that also conform to the RS-232 standard. Figure 2-5 shows an RS-232 connector.
- **V.35**: Typically used for modem-to-multiplexer communication, this ITU standard for high-speed, synchronous data exchange combines the bandwidth of several telephone

circuits. In the United States, V.35 is the interface standard used by most routers and DSUs that connect to T1 carriers. V.35 cables are high-speed serial assemblies designed to support higher data rates and connectivity between DTEs and DCEs over digital lines. You'll read more about DTEs and DCEs later in this section.

- **HSSI**: A High-Speed Serial Interface (HSSI) supports transmission rates of up to 52 Mbps. Engineers use HSSI to connect routers on LANs with WANs over high-speed lines such as T3 lines. Engineers also use HSSI to provide high-speed connectivity between LANs, using Token Ring or Ethernet. HSSI is a DTE/DCE interface developed by Cisco Systems and T3plus Networking to address the need for high-speed communication over WAN links.

Figure 2-5 Nine-Pin RS-232 Connector



As well as using different signaling methods, each of these standards uses different types of cables and connectors. Each standard plays a different role in a LAN-to-WAN topology. This course does not examine the details of V.35 and HSSI pinning schemes. However, a quick look at a nine-pin RS-232 connector used to connect a PC to a modem helps illustrate the concept, as shown in Figure 2-5. A later section looks at V.35 and HSSI cables.

- Pin 1: Data Carrier Detect (DCD) indicates that the carrier for the transmit data is ON.
- Pin 2: The receive pin (RD) carries data from the serial device to the computer.

- Pin 3: The transmit pin (TD) carries data from the computer to the serial device.
- Pin 4: ***Data Terminal Ready (DTR)*** tells the modem that the computer is ready to transmit.
- Pin 5: Ground.
- Pin 6: ***Data Set Ready (DSR)*** is similar to DTR. It indicates that the Data set is ON.
- Pin 7: The RTS pin requests clearance to send data to a modem.
- Pin 8: The serial device uses the ***Clear to Send (CTS)*** pin to acknowledge the computer's RTS signal. In most situations, RTS and CTS are constantly ON throughout the communication session.
- Pin 9: An auto-answer modem uses the Ring Indicator (RI) to signal receipt of a telephone ring signal.

The DCD and RI pins are available only in connections to a modem. These two lines are rarely used. Most modems transmit status information to a PC when a carrier signal is detected (when a connection is made to another modem) or when the modem receives a ring signal from the telephone line.

Note

The pinning scheme for the nine-pin RS-232 connector is included only as an example. If you are interested in a more complete explanation, consult other sources, such as ***http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html#anchor1154232***.

TDM

In the early 1960s, Bell Laboratories developed ***time-division multiplexing (TDM)*** to maximize the amount of voice traffic carried over a medium. Before multiplexing, each telephone call required its own physical link. This was an expensive and unscalable solution.

TDM is a signaling method that divides the bandwidth of a single link into separate channels or time slots. TDM transmits two or more channels over the same link by allocating a different time interval (time slot) for the transmission of each channel. In effect, the channels take turns using the link.

Time Division Multiplexing

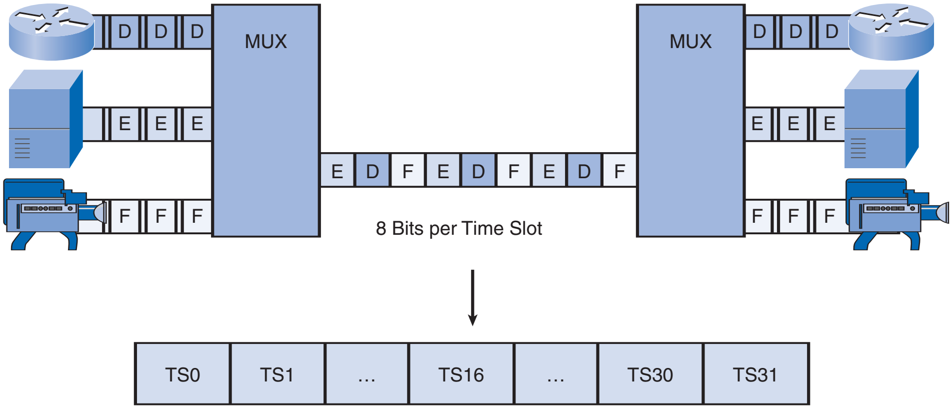
TDM is a physical layer concept. It has no regard for the nature of the information that is being multiplexed onto the output channel. TDM is independent of the Layer 2 protocol that is used by the input channels.

TDM can be explained by an analogy to highway traffic. To transport traffic from four roads to another city, you can send all the traffic on one highway lane if the feeding roads are equally serviced and the traffic is synchronized. So, if each of the four roads puts a car on the highway every four seconds, the highway receives one car each second. As long as the speed of all the cars is synchronized, no collisions occur. At the destination, the reverse happens: the cars are taken off the highway and are fed to the local roads by the same synchronous mechanism.

This is the principle used in synchronous TDM when sending data over a link. TDM increases the capacity of the ***transmission link*** by slicing time into smaller intervals so that the link carries the bits from multiple input sources. This effectively increases the number of bits transmitted per second. With TDM, the transmitter and receiver both know exactly which signal is being sent.

In Figure 2-6, a multiplexer (MUX) at the transmitter accepts three separate signals, depicted as D, E, and F. The MUX divides each signal into segments and puts each segment into a single channel by inserting each segment into a time slot. The figure shows multiple time slots, starting with time slot 0 (TS0).

Figure 2-6 TDM



- TDM shares available transmission time on a medium by assigning timeslots to users.
- The MUX accepts input from attached devices in a round-robin fashion and transmits the data in a never-ending pattern.
- T1/E1 and ISDN telephone lines are common examples of synchronous TDM.

A MUX at the receiving end reassembles the TDM stream into the three separate ***data streams*** based on only the timing of the arrival of each bit. A technique called bit interleaving keeps track of the number and sequence of the bits from each specific transmission so that they can be quickly and efficiently reassembled into their original form upon receipt. Byte interleaving performs the same functions, but because there are 8 bits in each byte, the process needs a bigger or longer time slot.

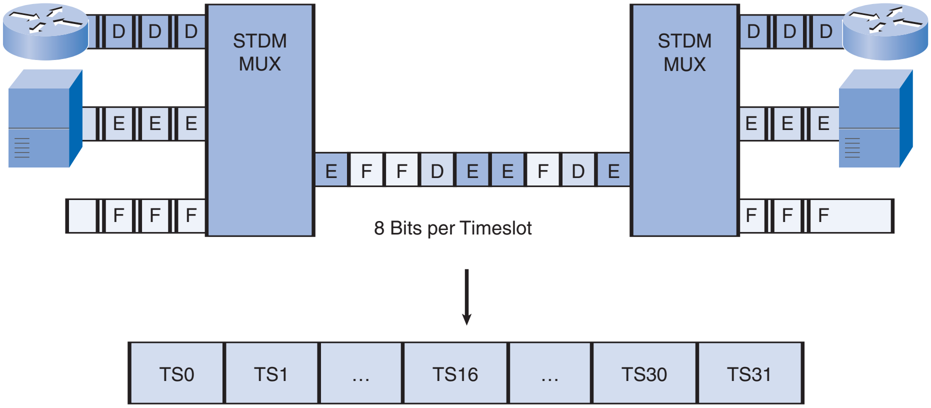
For more information on TDM, refer to ***http://www.networkdictionary.com/telecom/tdm.php***.

Statistical Time Division Multiplexing

In another analogy, compare TDM to a train with 32 railroad cars. Each car is owned by a different freight company, and every day the train leaves with the 32 cars attached. If one of the companies has cargo to send, that car is loaded. If the company has nothing to send, the car remains empty but stays on the train. Shipping empty containers is not very efficient. TDM shares this inefficiency when traffic is intermittent, because the time slot is still allocated even when the channel has no data to transmit.

Statistical time-division multiplexing (STDM) is a variation of TDM that was developed to overcome this inefficiency. STDM uses a variable time slot length, allowing channels to compete for any free slot space, as shown in Figure 2-7.

Figure 2-7 Statistical TDM

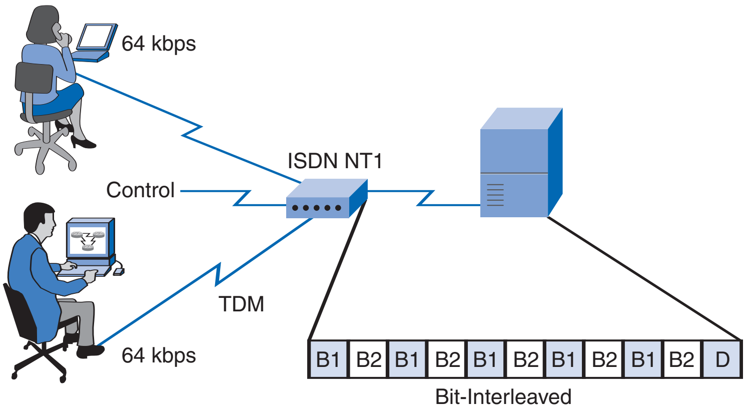


In the TDM example, signals D, E, and F were always sent in sequential order. Notice that in Figure 2-7, the signal transmissions are no longer sent sequentially. Instead, STDM embeds the signals as required into any available time slot. To do so, it employs ***buffer*** memory that temporarily stores the data during periods of peak traffic. STDM does not waste high-speed line time with inactive channels using this scheme. However, STDM requires each transmission to carry identification information (a channel identifier).

TDM Examples: ISDN and SONET

An example of a technology that uses synchronous TDM is ISDN. ISDN basic rate interface (BRI) has three channels: two 64-kbps B channels (B1 and B2) and a 16-kbps D channel. The TDM has ten time slots, which are repeated in the sequence shown in Figure 2-8.

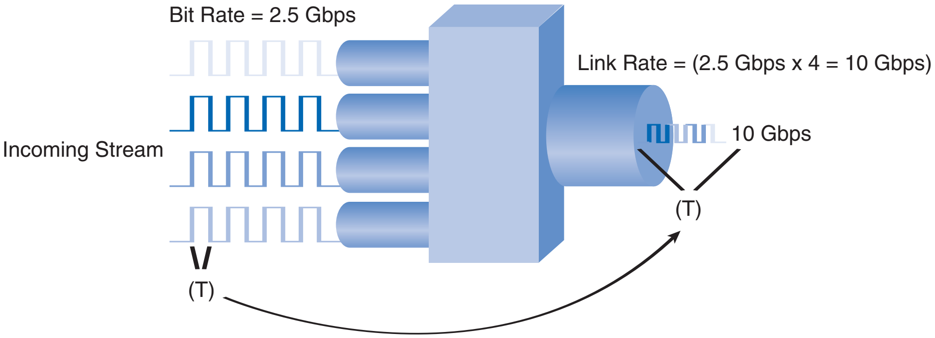
Figure 2-8 TDM Example: ISDN



On a larger scale, the telecommunications industry uses the ***SONET*** or SDH standard for optical transport of TDM data. SONET, used in North America, and SDH, used elsewhere, are two closely related standards that specify interface parameters, rates, framing formats, multiplexing methods, and management for synchronous TDM over fiber.

Figure 2-9 shows an example of statistical TDM. SONET/SDH takes *n* bit streams, multiplexes them, and optically modulates the signal, sending it out using a light-emitting device over fiber with a bit rate equal to (incoming bit rate) * *n*. Thus, traffic arriving at the SONET multiplexer from four places at 2.5 Gbps goes out as a single stream at 4 * 2.5 Gbps, or 10 Gbps. This principle is illustrated in the figure, which shows an increase in the bit rate by a factor of 4 in time slot T.

Figure 2-9 STDM Example: SONET



The original unit used in multiplexing telephone calls is 64 kbps, which represents one phone call. This is referred to as a ***DS0 (digital signal level zero)***. In North America, 24 DS0 units are multiplexed using TDM into a higher bit-rate signal with an aggregate speed of

1.544 Mbps for transmission over T1 lines. Outside North America, 32 DS0 units are multiplexed for E1 transmission at 2.048 Mbps.

Table 2-1 shows the signal level hierarchy for multiplexing telephone calls. As an aside, although it is common to refer to a 1.544-Mbps transmission as a T1, it is more correct to call it DS1.

Table 2-1 DS0 Units

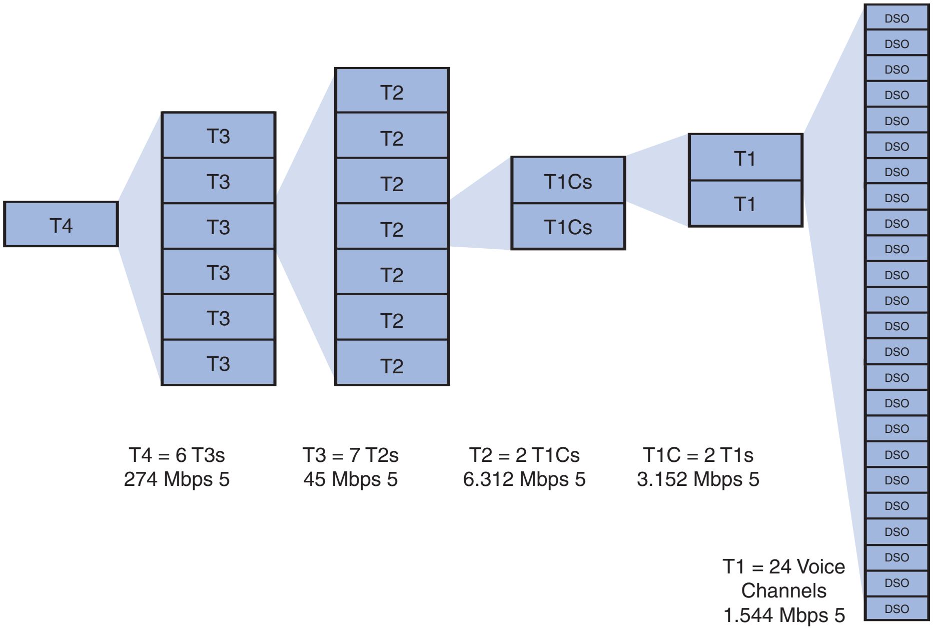
Signal Bit	Rate	Voice Slots
DS0	64 kbps	1 DS0
DS1	1.544 Mbps	24 DS0s
DS2	6.312 Mbps	96 DS0s
DS3	44.736 Mbps	672 DS0s or 28 DS1s

T-carrier refers to the bundling of DS0s. For example, a T1 equals 24 DSOs, a T1C equals 48 DSOs (or two T1s), and so on. Figure 2-10 shows a sample T-carrier infrastructure hierarchy. The E-carrier hierarchy is similar.

Note

For more information, refer to ***http://www.atis.org/tg2k/_t-carrier.html***.

Figure 2-10 T-Carrier Hierarchy



Demarcation Point

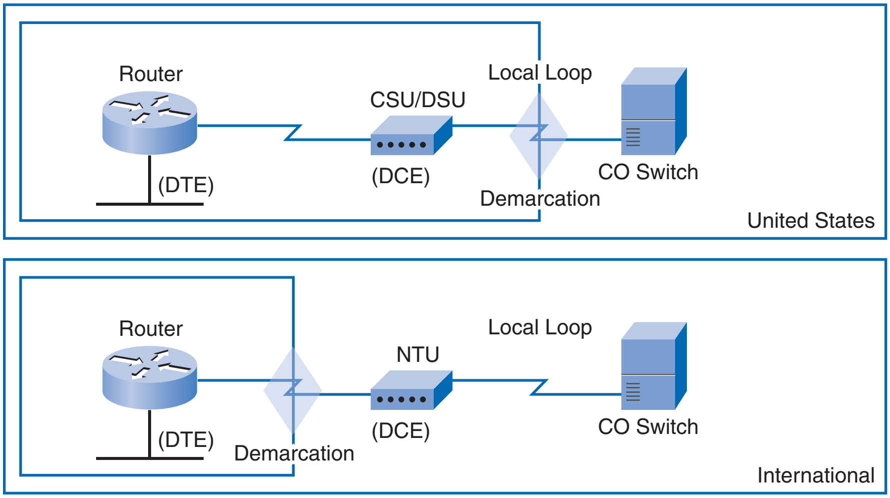
Before deregulation in North America and other countries, telephone companies owned the local loop, including the wiring and equipment on the premises of the customers. Deregulation forced telephone companies to unbundle their local loop infrastructure to allow other suppliers to provide equipment and services. This led to a need to delineate which part of the network the telephone company owned and which part the customer owned. This point of delineation is the demarcation point, or demarc.

Demarc

The demarcation point marks the point where your network interfaces with the network owned by another organization. In telephone terminology, this is the interface between customer premises equipment (CPE) and network service provider equipment. The demarcation point is the point in the network where the responsibility of the service provider ends.

Figure 2-11 shows an ISDN scenario. In the United States, a service provider provides the local loop to the customer premises, and the customer provides the active equipment such as the channel service unit/data service unit (CSU/DSU) on which the local loop is terminated. This termination often occurs in a telecommunications closet, and the customer is responsible for maintaining, replacing, or repairing the equipment.

Figure 2-11 Demarcation Point



In other countries, the network terminating unit (NTU) is provided and managed by the service provider. Therefore, the demarcation point is now located between the provider's NTU and the customer's router. The provider can now actively manage and troubleshoot the local loop. The customer connects a CPE device, such as a router or ***Frame Relay access device***, to the NTU using a V.35 or RS-232 serial interface.

Data Terminal Equipment and Data Communications Equipment

The term Data Terminal Equipment (DTE) refers to the device at the user end of the user-network interface. It serves as a data source, destination, or both. The DTE connects to a data network through Data Communications Equipment (DCE).

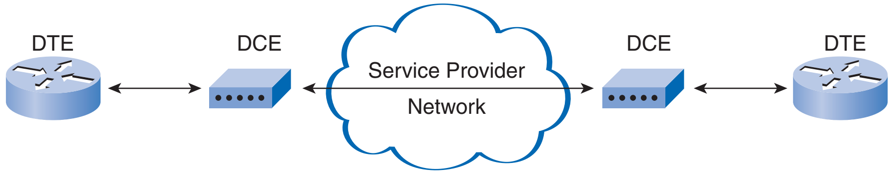
The DCE provides the physical connection to the network. It provides the clocking signal, which is used to synchronize the data transmission between the DCE and DTE, and forwards traffic.

DTE-DCE

From the point of view of connecting to the WAN, a serial connection has a DTE device at one end of the connection and a DCE device at the other end. The connection between the two DCE devices is the WAN service provider transmission network, as shown in Figure 2-12. In this case:

- The CPE, which generally is a router, is the DTE. The DTE could also be a terminal, computer, printer, or fax machine if they connect directly to the service provider network.
- The DCE, commonly a modem or CSU/DSU, is the device used to convert the user data from the DTE into a form acceptable to the WAN service provider transmission link. This signal is received at the remote DCE, which decodes the signal into a sequence of bits. The remote DCE then signals this sequence to the remote DTE.

Figure 2-12 Serial DCE and DTE Connections



The Electronics Industry Association (EIA) and the International Telecommunication Union Telecommunications Standardization Sector (ITU-T) have been most active in developing standards that allow DTEs to communicate with DCEs. The EIA calls the DCE data communications equipment, and the ITU-T calls the DCE data circuit-terminating equipment.

Cable Standards

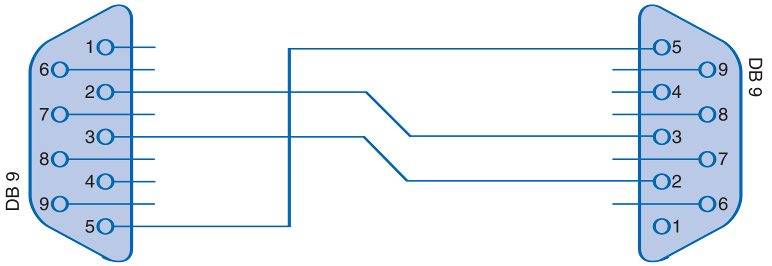
Originally, the concept of DCEs and DTEs was based on two types of equipment: terminal equipment that generated or received data, and communication equipment that only relayed data. In the development of the RS-232 standard, there were reasons why 25-pin RS-232 connectors on these two types of equipment needed to be wired differently. These reasons are no longer significant, but we are left with two different types of cables: one for connecting a DTE to a DCE, and another for connecting two DTEs directly to each other.

The DTE/DCE interface for a particular standard defines the following specifications:

- **Mechanical/physical**: The number of pins and connector type.
- **Electrical**: Defines voltage levels for 0 and 1.
- **Functional**: Specifies the functions that are performed by assigning meanings to each of the signaling lines in the interface.
- **Procedural**: Specifies the sequence of events for transmitting data.

The original RS-232 standard defined only the connection of DTEs with DCEs, which were modems. However, if you want to directly connect two DTEs, such as two computers or two routers, as we do in our labs, a special serial cable called a ***null modem*** eliminates the need for a DCE. It provides the same functionality as a crossover cable connected to the Ethernet interfaces of a PC. However, a null modem cable is used to connect the serial interfaces of the DTEs. Notice that with a null modem connection, the transmit (Tx) and receive (Rx) wires are crosslinked, as shown in Figure 2-13.

Figure 2-13 Null Modem to Connect Two DTEs



Connector 1	Connector 2	Function
2	2	Rx ⟵ Tx
3	3	Rx ⟶ Tx
5	5	Signal ground

Note the crosslinks: Pin 2 to Pin 3 and Pin 3 Pin 2

Note

A null modem cable also requires additional wires to be crossed. However, in this example we are simply focusing on the transmit and receive wires.

The cable for the DTE-to-DCE connection is a shielded serial transition cable. The router end of the shielded serial transition cable may be a DB-60 connector, which connects to the

DB-60 port on a serial WAN interface card. The other end of the serial transition cable is available with the connector appropriate for the standard that is to be used. The WAN provider or the CSU/DSU usually dictates this cable type. Cisco devices support the EIA/TIA-232, EIA/TIA-449, V.35, X.21, and EIA/TIA-530 serial standards, as shown in Figure 2-14.

Figure 2-14 WAN Serial Connection Options

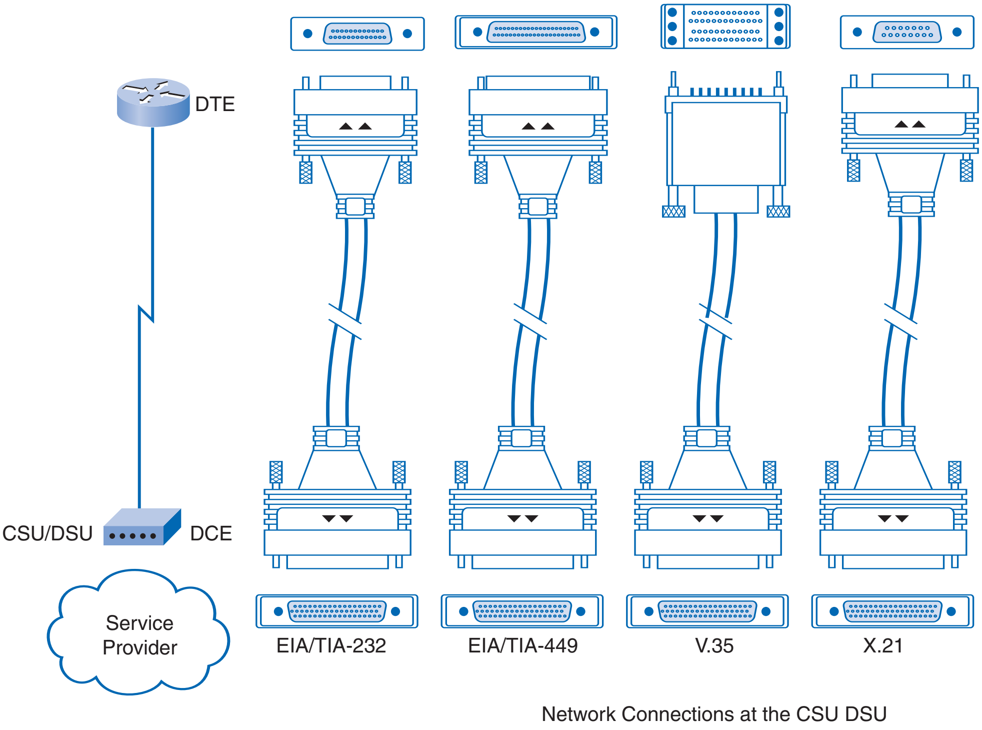
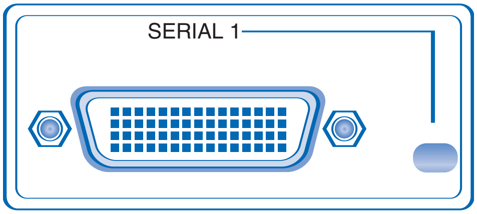


Figure 2-15 shows the DB-60 connector on a Cisco router.

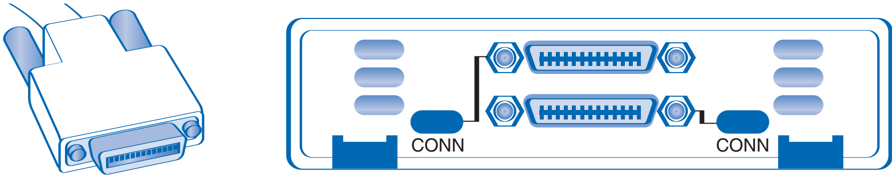
Figure 2-15 DB-60 Router Connector



To support higher port densities in a smaller form factor, Cisco has introduced a Smart Serial cable. The router interface end of the Smart Serial cable is a 26-pin connector that is significantly more compact than the DB-60 connector.

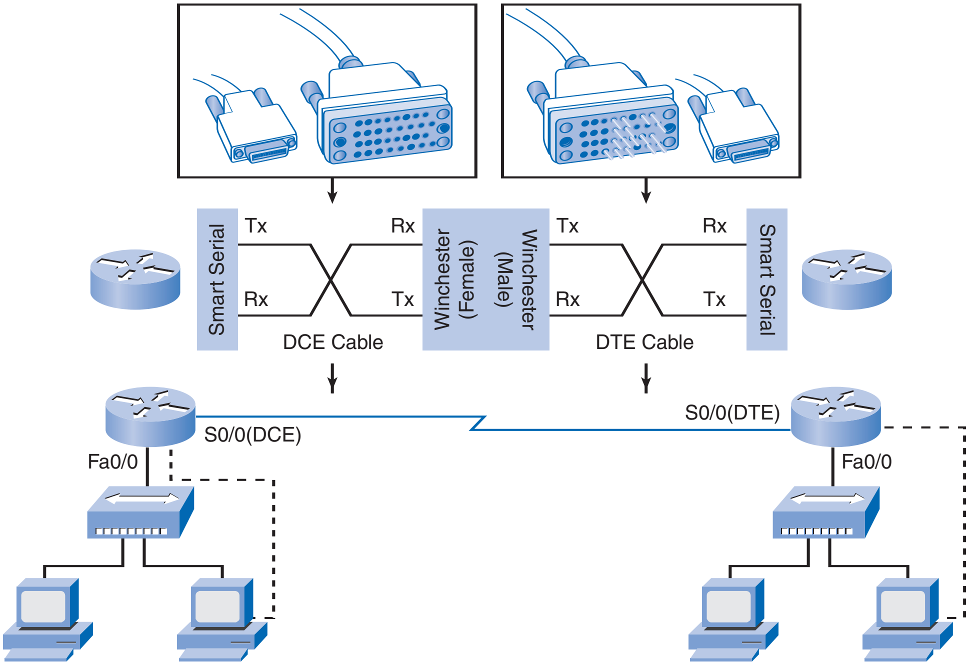
Figure 2-16 shows the Smart Serial cable and connection on a Cisco router. Notice that this connector contains two serial interfaces occupying the space that one DB-60 connector would take.

Figure 2-16 Smart Serial Router Connection



When using a null modem, keep in mind that synchronous connections require a clock signal. An external device can generate the signal, or one of the DTEs can generate the clock signal. When a DTE and DCE are connected, the serial port on a router is the DTE end of the connection by default, and the clock signal typically is provided by a CSU/DSU or similar DCE device, as shown in Figure 2-17. However, when you use a null modem cable in a router-to-router connection, one of the serial interfaces must be configured as the DCE end to provide the clock signal for the connection.

Figure 2-17 Serial WAN Connection in the Lab



Parallel-to-Serial Conversion

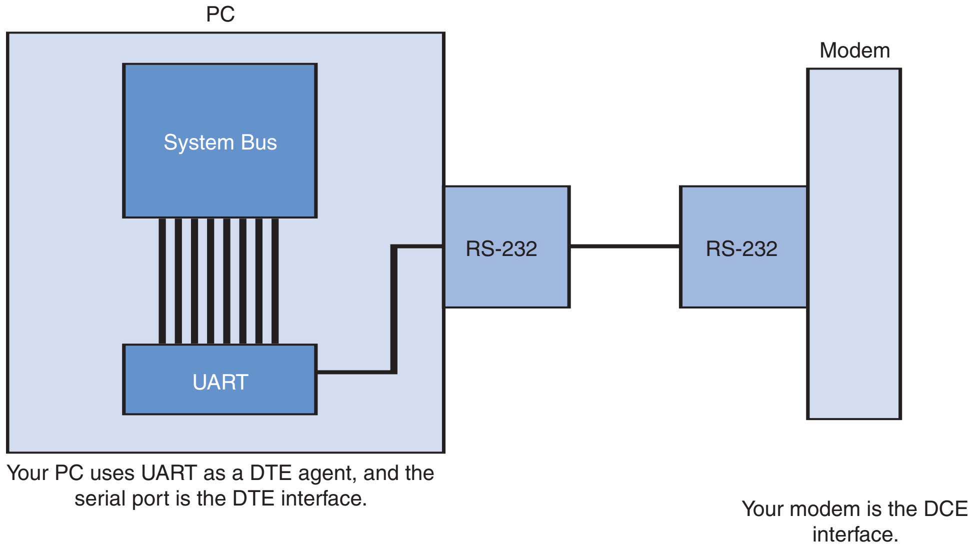
The terms DTE and DCE are relative with respect to what part of a network you are observing. RS-232C is the recommended standard (RS) describing the physical interface and protocol for relatively low-speed serial data communication between computers and related devices. The EIA originally defined RS-232C for teletypewriter devices in the early 1960s. The DTE is the RS-232C interface that a computer uses to exchange data with a modem or other serial device. The DCE is the RS-232C interface that a modem or other serial device uses in exchanging data with the computer.

Note

In the early 1990s, the EIA renamed the RS-232 standard EIA232. However, both terms are still acceptable when referring to the standard.

For instance, as shown in Figure 2-18, your PC typically uses an RS-232C interface to communicate and exchange data with connected serial devices such as a modem. Your PC also has a ***Universal Asynchronous Receiver/Transmitter (UART)*** chip on the motherboard. Because the data in your PC flows along parallel circuits, the UART chip converts the groups of bits in parallel to a serial stream of bits. To work faster, a UART chip has buffers so that it can cache data coming from the system bus while it processes data going out the serial port. The UART is the DTE agent of your PC. It communicates with the modem or other serial device, which, in accordance with the RS-232C standard, has a complementary interface called the DCE interface.

Figure 2-18 Parallel-to-Serial Conversion Example


PC
Modem
System Bus
RS-232
RS-232
UART
Your PC uses UART as a DTE agent, and the serial port is the DTE interface.
Your modem is the DCE interface.

HDLC Encapsulation

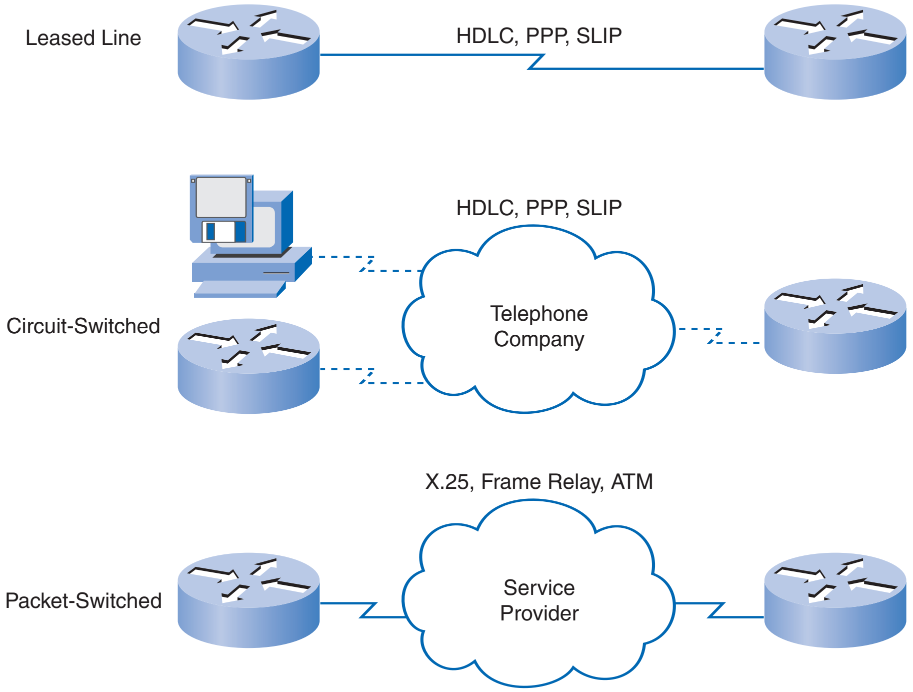
WANs use several different types of Layer 2 protocols, including PPP, Frame Relay, ATM, X.25, and HDLC. The HDLC protocol is introduced a little later.

Layer 2 WAN Encapsulation Protocols

On each WAN connection, data is encapsulated into frames before crossing the WAN link. To ensure that the correct protocol is used, you need to configure the appropriate Layer 2 encapsulation type. The choice of protocol depends on the WAN technology and the communicating equipment.

Figure 2-19 shows the more common WAN protocols and where they are used.

Figure 2-19 WAN Encapsulation Protocols



The following are short descriptions of the WAN protocols:

- **HDLC**: The default encapsulation type on point-to-point connections, dedicated links, and circuit-switched connections when the link uses two Cisco devices. HDLC is now the basis for synchronous PPP used by many servers to connect to a WAN, most commonly the Internet.

- **PPP**: Provides router-to-router and host-to-network connections over synchronous and asynchronous circuits. PPP works with several network layer protocols, such as IP and ***Internetwork Packet Exchange (IPX)***. PPP also has built-in security mechanisms such as PAP and CHAP. Most of this chapter deals with PPP.
- ***Serial Line Internet Protocol (SLIP)***: A standard protocol for point-to-point serial connections using TCP/IP. SLIP has been largely displaced by PPP.
- **X.25/*Link Access Procedure, Balanced (LAPB)***: An ITU-T standard that defines how connections between a DTE and DCE are maintained for remote terminal access and computer communications in public data networks. X.25 specifies LAPB, a data link layer protocol. X.25 is a predecessor to Frame Relay.
- **Frame Relay**: An industry-standard, switched, data link layer protocol that handles multiple virtual circuits. Frame Relay is a next-generation protocol after X.25. Frame Relay eliminates some of the overhead (such as error correction and flow control) employed in X.25. The next chapter is devoted to Frame Relay.
- **ATM**: The international standard for ***cell relay*** in which devices send multiple service types (such as voice, video, or data) in fixed-length (53-byte) cells. Fixed-length cells allow processing to occur in hardware, thereby reducing transit delays. ATM takes advantage of high-speed transmission media such as ***E3***, SONET, and T3.

HDLC Encapsulation

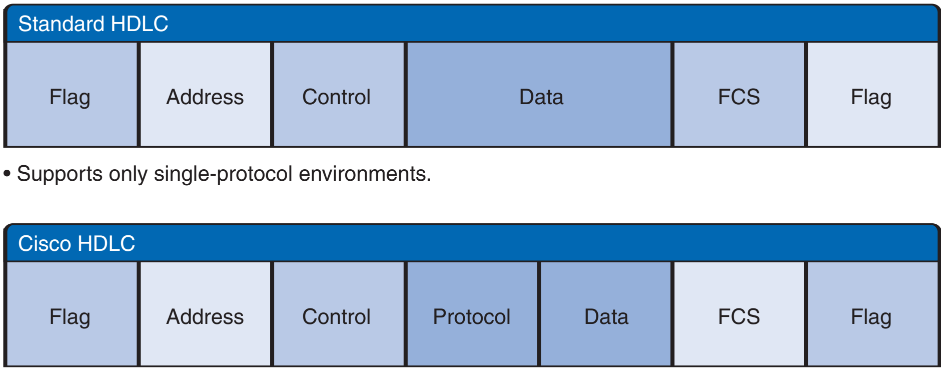
High-level Data Link Control (HDLC) is a ***bit-oriented*** synchronous data link layer protocol developed by the International Organization for Standardization (ISO). The current standard for HDLC is ISO 13239. HDLC was developed from the ***Synchronous Data Link Control (SDLC)*** standard proposed in the 1970s. HDLC provides both connection-oriented and connectionless service.

HDLC uses synchronous serial transmission to provide error-free communication between two points. HDLC defines a Layer 2 framing structure that allows for flow control and error control through the use of acknowledgments. Each frame has the same format, whether it is a data frame or a control frame.

When you want to transmit frames over synchronous or asynchronous links, you must remember that those links have no mechanism to mark the beginnings or ends of frames. HDLC uses a frame delimiter, or flag, to mark the beginning and end of each frame.

Cisco has developed an extension to the HDLC protocol to solve the inability to provide multiprotocol support. In fact, HDLC is the default encapsulation protocol on all Cisco serial interfaces. Although Cisco HDLC (also called cHDLC) is proprietary, Cisco has allowed many other network equipment vendors to implement it. Cisco HDLC frames contain a field for identifying the network protocol being encapsulated. Figure 2-20 compares HDLC to Cisco HDLC.

Figure 2-20 Standard and Cisco HDLC Frame Format



- Supports only single-protocol environments.
- Uses a protocol data field to support multiprotocol environments.

HDLC defines three types of frames, each with a different Control field format. The following descriptions summarize the fields illustrated in the figure:

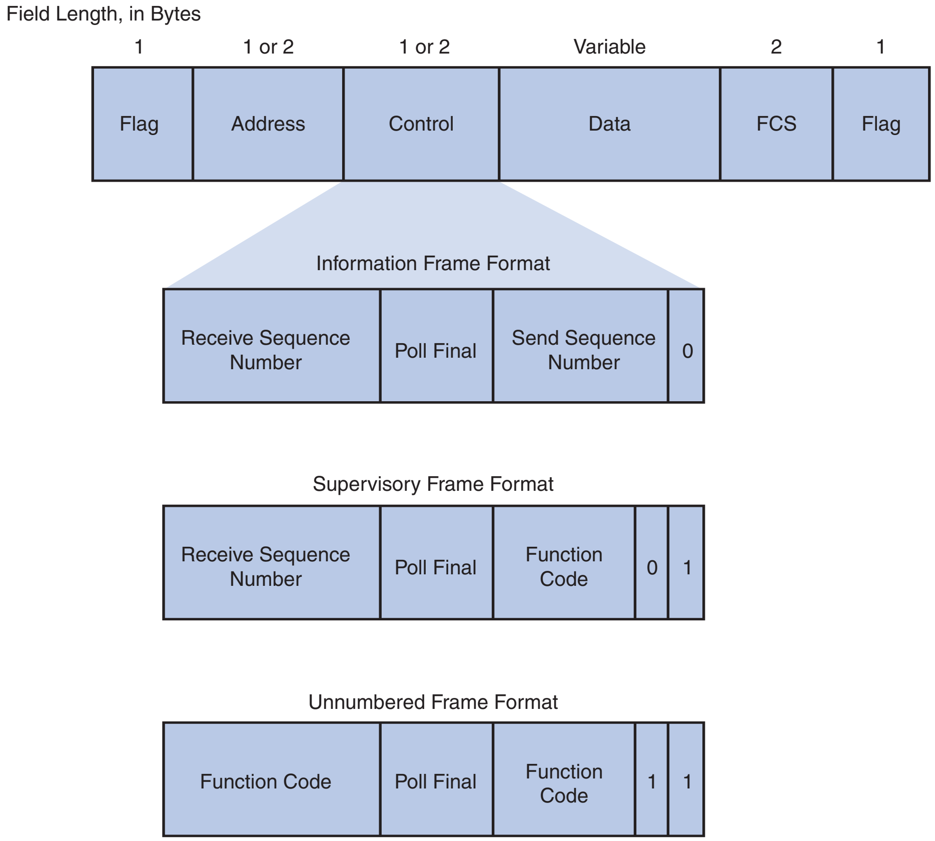
- **Flag**: The Flag field initiates and terminates error checking. The frame always starts and ends with an 8-bit Flag field. The bit pattern is 01111110. Because there is a likelihood that this pattern occurs in the actual data, the sending HDLC system always inserts a 0 bit after every five 1s in the Data field, so in practice the flag sequence can occur only at the frame ends. The receiving system strips out the inserted bits. When frames are transmitted consecutively, the end flag of the first frame is used as the start flag of the next frame.
- **Address**: The HDLC standard can be configured in point-to-point and multipoint connections. In point-to-point HDLC connections, this field is empty.
- **Control**: The Control field uses three different formats, depending on the type of HDLC frame used: Information, Supervisory, and Unnumbered frames.
- **Protocol** (used only in Cisco HDLC): This field specifies the protocol type encapsulated within the frame (such as 0x0800 for IP).
- **Data**: A variable-length field that contains Layer 3 packets.
- **Frame check sequence (FCS)**: The FCS precedes the ending flag delimiter and usually is a cyclic redundancy check (CRC) calculation remainder. The CRC calculation is redone in the receiver. If the result differs from the value in the original frame, an error is assumed.

Figure 2-21 and the following list summarize the control frames:

- **Information (I) frame**: I-frames carry upper-layer information and some control information. This frame sends and receives sequence numbers, and the poll final (P/F) bit performs flow and error control. The send sequence number refers to the number of the frame to be sent next. The receive sequence number provides the number of the frame to be received next. Both sender and receiver maintain send and receive sequence numbers. A ***primary station*** uses the P/F bit to tell the secondary whether it requires an immediate response. A secondary station uses the P/F bit to tell the primary whether the current frame is the last in its current response.

- **Supervisory (S) frame**: S-frames provide control information. An S-frame can request and suspend transmission, report on status, and acknowledge receipt of I-frames. S-frames do not have an Information field.
- **Unnumbered (U) frame**: U-frames support control purposes and are not sequenced. A U-frame can be used to initialize secondaries. Depending on the function of the U-frame, its Control field is 1 or 2 bytes. Some U-frames have an Information field.

Figure 2-21 HDLC Frame Types



Configuring HDLC Encapsulation

Cisco HDLC is the default encapsulation method used by Cisco devices on synchronous serial lines.

You use Cisco HDLC as a point-to-point protocol on leased lines between two Cisco devices. If you are connecting to a non-Cisco device, use synchronous PPP.

If the default encapsulation method has been changed, use the **encapsulation hdlc** command in privileged mode to reenable HDLC.

There are two steps to enable HDLC encapsulation:

Step 1. Enter the interface configuration mode of the serial interface.

Step 2. Enter the **encapsulation hdlc** command to specify the encapsulation protocol on the interface.

To configure HDLC on a serial interface, you would enter the following configuration:

```
R1(config)# interface serial 0/0/0
R1(config-router)# encapsulation hdlc
```

Troubleshooting Serial Interfaces

The output of the **show interfaces serial** command displays information specific to serial interfaces. When HDLC is configured, "Encapsulation HDLC" should be reflected in the output, as highlighted in Example 2-1.

Example 2-1 Verifying a Serial PPP Encapsulation Configuration

```
R1# show interfaces serial 0/0/0

Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.0.1/30
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:03, output 00:00:04, output hang never
  Last clearing of "show interface" counters 1w0d
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     219 packets input, 15632 bytes, 0 no buffer
     Received 218 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     217 packets output, 14919 bytes, 0 underruns
     0 output errors, 0 collisions, 107 interface resets
     0 output buffer failures, 0 output buffers swapped out
     12 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

The **show interfaces serial** command returns one of six possible states. You can see any of the following possible states in the interface status line:

- Serial *x* is up, line protocol is up
- Serial *x* is down, line protocol is down
- Serial *x* is up, line protocol is down
- Serial *x* is up, line protocol is up (looped)
- Serial *x* is up, line protocol is down (disabled)
- Serial *x* is administratively down, line protocol is down

Table 2-2 explains some of the possible conditions and solutions.

Table 2-2 Troubleshooting a Serial Interface

Status Line	Possible Condition	Problem/Solution
Serial *x* is up, line protocol is up	This is the proper status line condition.	No action is required.
Serial *x* is down, line protocol is down (DTE mode)	The router is not sensing a CD signal, which means the CD is not active. A WAN carrier service provider problem has occurred, which means the line is down or is not connected to the CSU/DSU. Cabling is faulty or incorrect. Hardware failure has occurred (CSU/DSU).	1. Check the LEDs on the CSU/DSU to see whether the CD is active, or insert a breakout box on the line to check for the CD signal. 2. Verify that the proper cable and interface are being used by looking at the hardware installation documentation. 3. Insert a breakout box, and check all control leads. 4. Contact the leased-line or other carrier service to see whether there is a problem. 5. Swap faulty parts. 6. If faulty router hardware is suspected, change the serial line to another port. If the connection comes up, the previously connected interface has a problem.

continues

Table 2-2 Troubleshooting a Serial Interface

Status Line	Possible Condition	Problem/Solution
Serial *x* is up, line protocol is down (DTE mode)	A local or remote router is misconfigured. The remote router is not sending keepalives. A leased-line or other carrier service problem has occurred, which means a noisy line or misconfigured or failed switch. A timing problem has occurred on the cable, which means serial clock transmit external (SCTE) is not set on the CSU/DSU. SCTE is designed to compensate for clock phase shift on long cables. When the DCE device uses SCTE instead of its internal clock to sample data from the DTE, it is better able to sample the data without error, even if there is a phase shift in the cable. A local or remote CSU/DSU has failed. Router hardware, which could be either local or remote, has failed.	1. Put the modem, CSU, or DSU in local loopback mode, and use the **show interfaces serial** command to see whether the line protocol comes up. If it does, a WAN carrier service provider problem or a failed remote router is the likely problem. 2. If the problem appears to be on the remote end, repeat Step 1 on the remote modem, CSU, or DSU. 3. Verify all cabling. Make certain that the cable is attached to the correct interface, the correct CSU/DSU, and the correct WAN carrier service provider network termination point. Use the **show controllers** EXEC command to determine which cable is attached to which interface. 4. Enable the **debug serial interface** EXEC command. 5. If the line protocol does not come up in local loopback mode, and if the output of the **debug serial interface** EXEC command shows that the keepalive counter is not incrementing, a router hardware problem is likely. Swap the router interface hardware.

Table 2-2 Troubleshooting a Serial Interface

Status Line	Possible Condition	Problem/Solution
		6. If the line protocol comes up and the keepalive counter increments, the problem is not in the local router. 7. If faulty router hardware is suspected, change the serial line to an unused port. If the connection comes up, the previously connected interface has a problem.
Serial *x* is up, line protocol is down (DCE mode)	The **clockrate** interface configuration command is missing. The DTE device does not support or is not set up for SCTE mode (terminal timing). The remote CSU or DSU has failed.	1. Add the **clockrate** interface configuration command on the serial interface. Syntax: **clockrate** *bps* Syntax description: *bps* is the desired clock rate in bits per second: 1200, 2400, 4800, 9600, 19200, 38400, 56000, 64000, 72000, 125000, 148000, 250000, 500000, 800000, 1000000, 1300000, 2000000, 4000000, or 8000000. 2. If the problem appears to be on the remote end, repeat Step 1 on the remote modem, CSU, or DSU. 3. Verify that the correct cable is being used. 4. If the line protocol is still down, there is a possible hardware failure or cabling problem. Insert a breakout box, and observe the leads. 5. Replace faulty parts as necessary.

continues

Table 2-2 Troubleshooting a Serial Interface

Status Line	Possible Condition	Problem/Solution
Serial *x* is up, line protocol is up (looped)	A loop exists in the circuit. The sequence number in the keepalive packet changes to a random number when a loop is initially detected. If the same random number is returned over the link, a loop exists.	1. Use the **show running-config** privileged EXEC command to look for any loopback interface configuration command entries. 2. If there is a **loopback** interface configuration command entry, use the **no loopback** interface configuration command to remove the loop. 3. If there is no **loopback** interface configuration command, examine the CSU/DSU to determine whether they are configured in manual loopback mode. If they are, disable manual loopback. 4. After disabling loopback mode on the CSU/DSU, reset the CSU/DSU, and inspect the line status. If the line protocol comes up, no other action is needed. 5. If, upon inspection, the CSU or DSU cannot be manually set, contact the leased-line or other carrier service for line troubleshooting assistance.

Table 2-2 Troubleshooting a Serial Interface

Status Line	Possible Condition	Problem/Solution
Serial *x* is up, line protocol is down (disabled)	A high error rate has occurred due to a WAN service provider problem. A CSU or DSU hardware problem has occurred. Router hardware (interface) is bad.	1. Troubleshoot the line with a serial analyzer and breakout box. Look for toggling CTS and DSR signals. 2. Loop the CSU/DSU (DTE loop). If the problem continues, it is likely that there is a hardware problem. If the problem does not continue, it is likely that there is a WAN service provider problem. 3. Swap out bad hardware as required (CSU, DSU, switch, local or remote router).
Serial *x* is administratively down, line protocol is down	The router configuration includes the **shutdown** interface configuration command. A duplicate IP address exists.	1. Check the router configuration for the **shutdown** command. 2. Use the **no shutdown** interface configuration command to remove the **shutdown** command. 3. Verify that no identical IP addresses are using the **show running-config** privileged EXEC command or the **show interfaces** EXEC command. 4. If there are duplicate addresses, resolve the conflict by changing one of the IP addresses.

The **show controllers** command is another important diagnostic tool when you're troubleshooting serial lines. The output indicates the state of the interface channels and whether a cable is attached to the interface.

In Example 2-2, serial interface 0/0/0 has a V.35 DCE cable attached. The command syntax varies, depending on the platform. *Cisco 7000* series routers use a cBus controller card to connect serial links. With these routers, use the **show controllers cbus** command.

Example 2-2 Verifying a Serial PPP Encapsulation Configuration

```
R1# show controllers serial 0/0/0

Interface Serial0/0/0
Hardware is GT96K
DCE V.35, clock rate 64000
idb at 0x62938244, driver data structure at 0x6293A608
wic_info 0x6293AC04
Physical Port 0, SCC Num 0
MPSC Registers:
MMCR_L=0x000304C0, MMCR_H=0x00000000, MPCR=0x00000000
CHR1=0x00FE007E, CHR2=0x00000000, CHR3=0x000005F4, CHR4=0x00000000
CHR5=0x00000000, CHR6=0x00000000, CHR7=0x00000000, CHR8=0x00000000
CHR9=0x00000000, CHR10=0x00003008
SDMA Registers:
SDC=0x00002201, SDCM=0x00000080, SGC=0x0000C000
CRDP=0x073BD020, CTDP=0x073BD450, FTDB=0x073BD450
Main Routing Register=0x00038E00 BRG Conf Register=0x0005023F
Rx Clk Routing Register=0x76583888 Tx Clk Routing Register=0x76593910
GPP Registers:
Conf=0x43430002, Io=0x4646CA50, Data=0x7F6B3FAD, Level=0x80004
Conf0=0x43430002, Io0=0x4646CA50, Data0=0x7F6B3FAD, Level0=0x80004
0 input aborts on receiving flag sequence
0 throttles, 0 enables
0 overruns
0 transmitter underruns
 --More--
```

If the electrical interface output is shown as UNKNOWN instead of V.35, EIA/TIA-449, or some other electrical interface type, the likely problem is an improperly connected cable. A problem with the card's internal wiring is also possible. If the electrical interface is unknown, the corresponding display for the **show interfaces serial** *x* command shows that the interface and line protocol are down.

Packet Tracer
□ **Activity**

Troubleshooting a Serial Interface 2.1.7

In this activity, you practice troubleshooting serial interfaces. Detailed instructions are provided within the activity. Use File e4-217.pka on the CD-ROM that accompanies this book to perform this activity using Packet Tracer.

PPP Concepts

This section examines Point-to-Point Protocol (PPP) in more detail.

Introducing PPP

Recall that HDLC is the default serial encapsulation method when you connect two Cisco routers. With an added Protocol Type field, the Cisco version of HDLC is proprietary. Thus, Cisco HDLC can work only with other Cisco devices. However, when you need to connect to a non-Cisco router, you should use PPP encapsulation.

What Is PPP?

PPP encapsulation has been carefully designed to retain compatibility with most commonly used supporting hardware. PPP encapsulates data frames for transmission over physical links. PPP establishes a direct connection using serial cables, phone lines, ***trunk*** lines, cellular telephones, specialized radio links, or fiber-optic links. Using PPP has many advantages, including the fact that it is not proprietary. Moreover, it includes many features not available in HDLC:

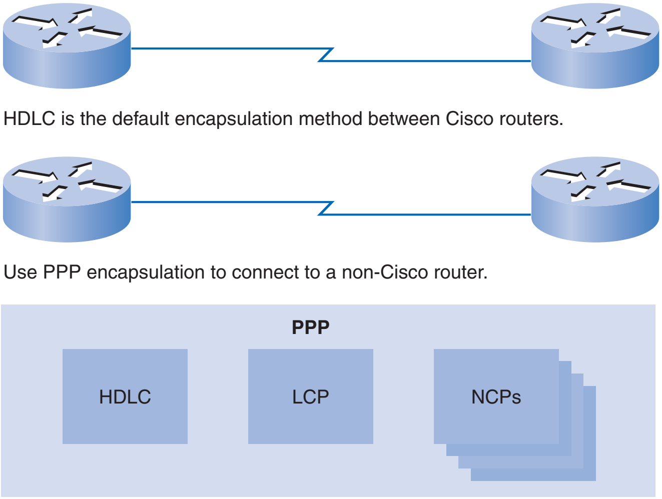
- The link quality management feature monitors the quality of the link. If too many errors are detected, PPP takes the link down.
- PPP supports PAP and CHAP authentication. This feature is explained and practiced in a later section.

PPP contains three main components:

- It uses the HDLC protocol as a basis for encapsulating datagrams over point-to-point links.
- It uses an extended version of ***Link Control Protocol (LCP)*** to establish, configure, and test the data link connection.
- It has a family of ***Network Control Protocols (NCP)*** to establish and configure different network layer protocols. PPP allows the simultaneous use of multiple network layer protocols. Some of the more common NCPs are Internet Protocol Control Protocol, AppleTalk Control Protocol, ***Novell IPX*** Control Protocol, Cisco Systems Control Protocol, ***SNA Control Protocol***, and Compression Control Protocol.

Figure 2-22 shows the default HDLC encapsulation on Cisco routers, as well as the more common PPP encapsulation and its three main components.

Figure 2-22 What Is PPP?


HDLC is the default encapsulation method between Cisco routers.
Use PPP encapsulation to connect to a non-Cisco router.
PPP
HDLC
LCP
NCPs

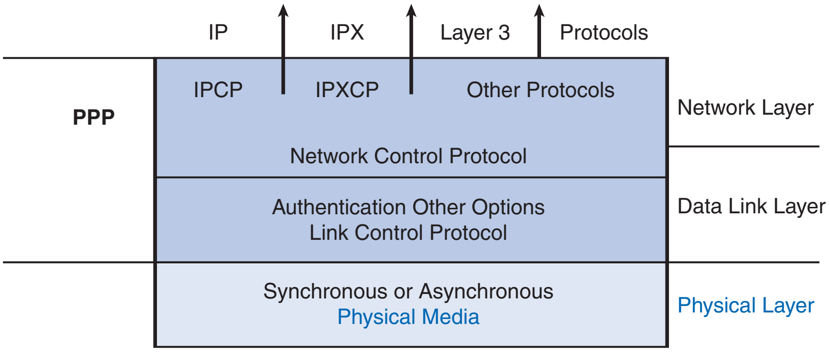
PPP Layered Architecture

A layered architecture is a logical model, design, or blueprint that aids communication between interconnecting layers.

PPP Architecture

Figure 2-23 maps the layered architecture of PPP against the Open Systems Interconnection (OSI) model. PPP and OSI share the same physical layer, but PPP distributes the functions of LCP and NCP differently.

Figure 2-23 PPP Layered Architecture: Physical Layer



With its lower level functions, PPP can use the following:

- Synchronous physical media
- Asynchronous physical media like those that use basic telephone service for modem dialup connections

At the physical layer, you can configure PPP on a range of interfaces:

- Asynchronous serial
- Synchronous serial
- HSSI
- ISDN

PPP operates across any DTE/DCE interface (RS-232-C, RS-422, RS-423, or V.35). The only absolute requirement imposed by PPP is a duplex circuit, either dedicated or switched, that can operate in either an asynchronous or synchronous bit-serial mode, transparent to PPP link layer frames. PPP does not impose any transmission rate restrictions other than those imposed by the particular DTE/DCE interface in use.

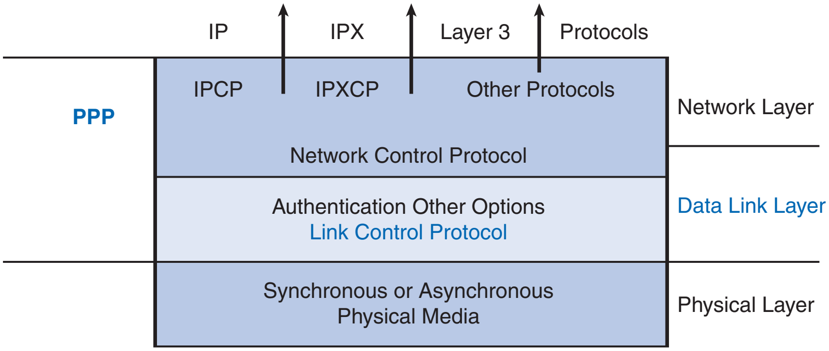
Most of the work done by PPP is at the data link and network layers by the LCP and NCPs. The LCP sets up the PPP connection and its parameters, the NCPs handle higher-layer protocol configurations, and the LCP terminates the PPP connection.

PPP Architecture: Link Control Protocol Layer

The LCP is the real working part of PPP. The LCP sits on top of the physical layer and has a role in establishing, configuring, and testing the data-link connection between devices.

The LCP, as shown in Figure 2-24, establishes the point-to-point link. The LCP also negotiates and sets up control options on the WAN data link, which are handled by the NCPs.

Figure 2-24 PPP Layered Architecture: LCP Layer



PPP offers service options in LCP and is primarily used for negotiation and frame checking when implementing the point-to-point controls specified by an administrator.

The LCP provides automatic configuration of the interfaces at each end, including the following:

- Handling varying limits on packet size
- Detecting common misconfiguration errors
- Terminating the link
- Determining when a link is functioning properly or when it is failing

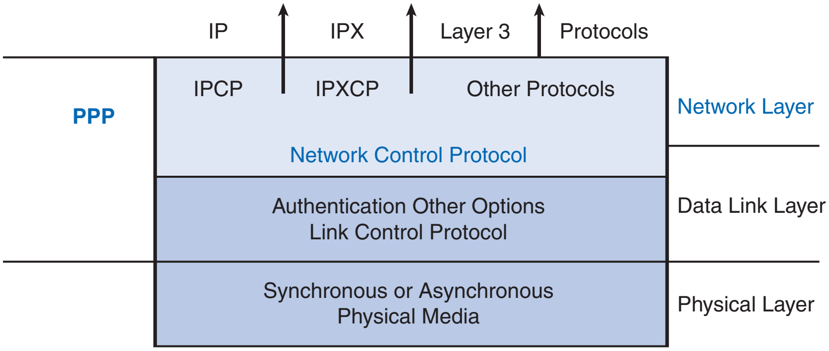
PPP also uses the LCP to agree automatically on encapsulation formats (authentication, compression, error detection) as soon as the link is established.

PPP Architecture: Network Control Protocol Layer

Point-to-point links tend to worsen many problems with the current family of network protocols. For instance, assigning and managing IP addresses, which is a problem even in LAN environments, is especially difficult over circuit-switched point-to-point links (such as dialup modem servers). PPP addresses these issues using NCPs.

PPP permits multiple network layer protocols to operate on the same communications link. For every network layer protocol used, PPP uses a separate NCP, as shown in Figure 2-25. For example, IP uses the IP Control Protocol (IPCP), and IPX uses the Novell IPX Control Protocol (IPXCP).

Figure 2-25 PPP Architecture: Network Layer



With its higher level functions, PPP carries packets from several network layer protocols in NCPs. These are functional fields containing standardized codes to indicate the network layer protocol type that PPP encapsulates.

NCPs include functional fields containing standardized codes (PPP protocol field numbers, as shown in Table 2-3) to indicate the network layer protocol that PPP encapsulates.

Table 2-3 NCPs

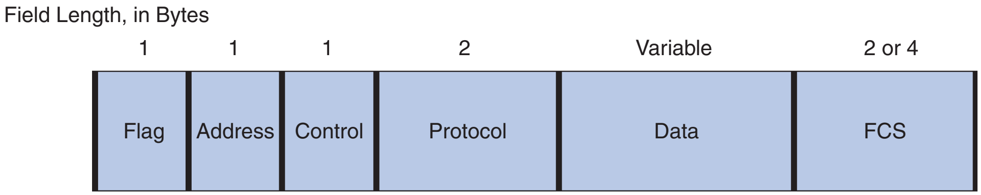
Value (in Hex)	Protocol Name
8021	Internet Protocol Control Protocol
8023	OSI Network Layer Control Protocol
8029	AppleTalk Control Protocol
802b	Novell IPX Control Protocol
C021	Link Control Protocol
C023	Password Authentication Protocol
C223	Challenge Handshake Authentication Protocol

Each NCP manages the specific needs of its respective network layer protocols. The various NCP components encapsulate and negotiate options for multiple network layer protocols. Using NCPs to configure the various network layer protocols is explained and practiced later in this chapter.

PPP Frame Structure

A PPP frame has six fields, as shown in Figure 2-26.

Figure 2-26 PPP Frame Fields



The fields in the PPP frame contain the following information:

- **Flag**: Indicates the beginning or end of a frame. Consists of the binary sequence 01111110 to identify a PPP frame. The value is set to 0x7E (bit sequence 011111110) to signify the start and end of a PPP frame. In successive PPP frames, only a single flag character is used.
- **Address**: Consists of the standard broadcast address, which is the binary sequence 11111111. PPP does not assign individual station addresses.
- **Control**: 1 byte that consists of the binary sequence 00000011, which calls for transmission of user data in an unsequenced frame. This provides a connectionless link service that does not require you to establish data links or link stations. In HDLC environments,

the Address field is used to address the frame to the destination node. On a point-to-point link, the destination node does not need to be addressed. Therefore, for PPP, the Address field is set to 0xFF, the broadcast address. If both PPP peers agree to perform Address and Control field compression during LCP negotiation, the Address field is not included.

- **Protocol**: As shown in Figure 2-26, the Protocol field consists of 2 bytes that identify the protocol encapsulated in the frame's Data field. The 2-byte Protocol ID field identifies the protocol of the PPP payload. If both PPP peers agree to perform Protocol field compression during LCP negotiation, the Protocol ID field is 1 byte for Protocol IDs in the range 0x00-00 to 0x00-FF.
- **Data**: 0 or more bytes that contain the datagram for the protocol specified in the Protocol field. The 2 bytes of the Frame Check Sequence (FCS) field, followed by the closing flag (not displayed), mark the end of the Data field. The default maximum length of the Data field is 1500 bytes.
- **Frame Check Sequence (FCS)**: A 16-bit checksum that is used to check for bit-level errors in the PPP frame. If the receiver's calculation of the FCS does not match the FCS in the PPP frame, the PPP frame is silently discarded. By prior agreement, consenting PPP implementations can use a 32-bit (4-byte) FCS for improved error detection.

The LCP can negotiate modifications to the standard PPP frame structure.

Establishing a PPP Session

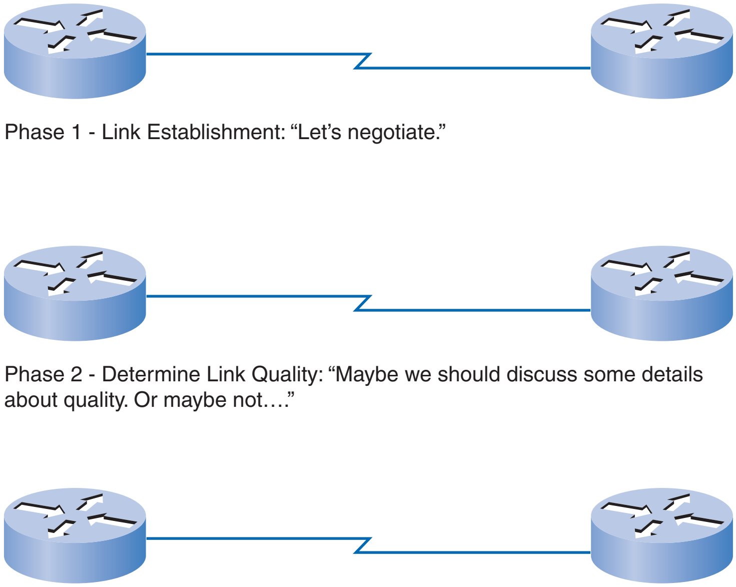
Establishing a PPP session consists of three phases performed by the LCP.

Establishing a PPP Session

Figure 2-27 shows the three phases of establishing a PPP session:

- **Phase 1: Link establishment and configuration negotiation**: Before PPP exchanges any network layer datagrams (such as IP), the LCP must open the connection and negotiate configuration options. This phase is complete when the receiving router sends a configuration-acknowledgment frame back to the router, initiating the connection.
- **Phase 2: Link quality determination (optional)**: The LCP tests the link to determine whether the link quality is sufficient to bring up network layer protocols. The LCP can delay transmission of network layer protocol information until this phase is complete.
- **Phase 3: Network layer protocol configuration negotiation**: After the LCP has finished the link quality determination phase, the appropriate NCP can separately configure the network layer protocols, and bring them up and take them down at any time. If the LCP closes the link, it informs the network layer protocols so that they can take appropriate action.

Figure 2-27 Establishing a PPP Session



Phase 1 - Link Establishment: "Let's negotiate."

Phase 2 - Determine Link Quality: "Maybe we should discuss some details about quality. Or maybe not…."

Phase 3 - Network Protocol Negotiation: "Okay, I will leave it to the NCPs to discuss higher-level details."

The LCP does all the talking.

The link remains configured for communications until explicit LCP or NCP frames close the link, or until some external event occurs (for example, an inactivity timer expires or a user intervenes). The LCP can terminate the link at any time. This usually is done when one of the routers requests termination, but it can happen because of a physical event, such as the loss of a carrier or the expiration of an idle-period timer.

Establishing a Link with LCP

LCP operation includes provisions for link establishment, link maintenance, and link termination.

LCP Operation

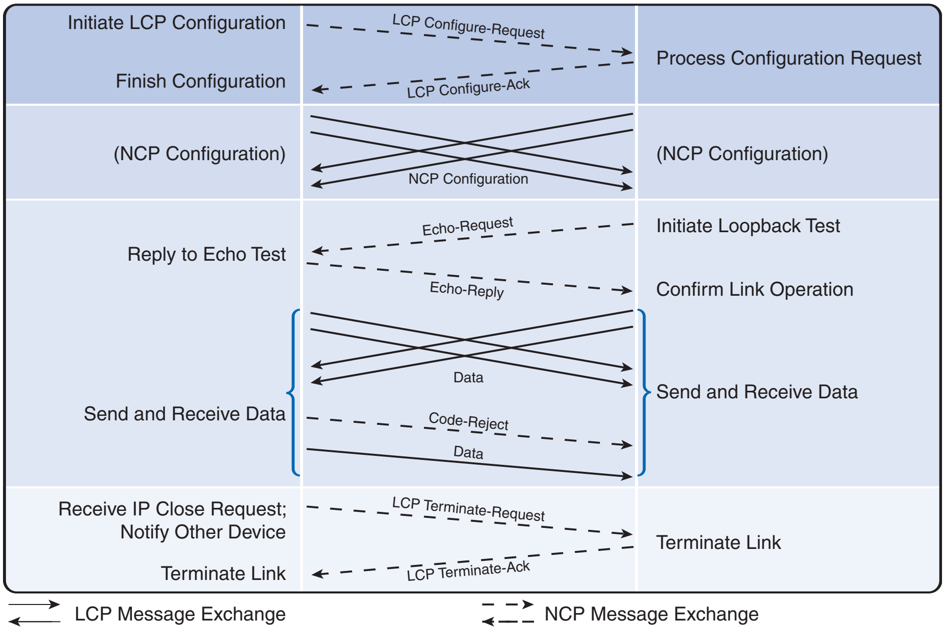
LCP operation uses three classes of LCP frames to accomplish the work of each LCP phase:

- Link-establishment frames establish and configure a link (Configure-Request, Configure-Ack, Configure-Nak, and Configure-Reject).

- Link-maintenance frames manage and debug a link (Code-Reject, Protocol-Reject, Echo-Request, Echo-Reply, and Discard-Request).
- Link-termination frames terminate a link (Terminate-Request and Terminate-Ack).

Figure 2-28 shows the LCP process. The first phase of LCP operation is link establishment. This phase must complete successfully before any network layer packets can be exchanged. During link establishment, the LCP opens the connection and negotiates the configuration parameters.

Figure 2-28 Establishing the Link



The link establishment process starts with the initiating device sending a Configure-Request frame to the responder. The Configure-Request frame includes a variable number of configuration options needed to set up on the link. In other words, the initiator sends a “wish list” to the responder.

The initiator’s wish list includes options for how it wants the link created, including protocol or authentication parameters. The responder processes the wish list. If it is acceptable, the responder responds with a Configure-Ack message. After receiving the Configure-Ack message, the process moves on to the authentication stage.

If the options are unacceptable or unrecognized, the responder sends a Configure-Nak or Configure-Reject. If a Configure-Ack is received, the operation of the link is handed over to the NCP. If either a Configure-Nak or Configure-Reject message is sent to the requester, the link is not established. If the negotiation fails, the initiator needs to restart the process with new options.

During link maintenance, LCP can use messages to provide feedback and test the link:

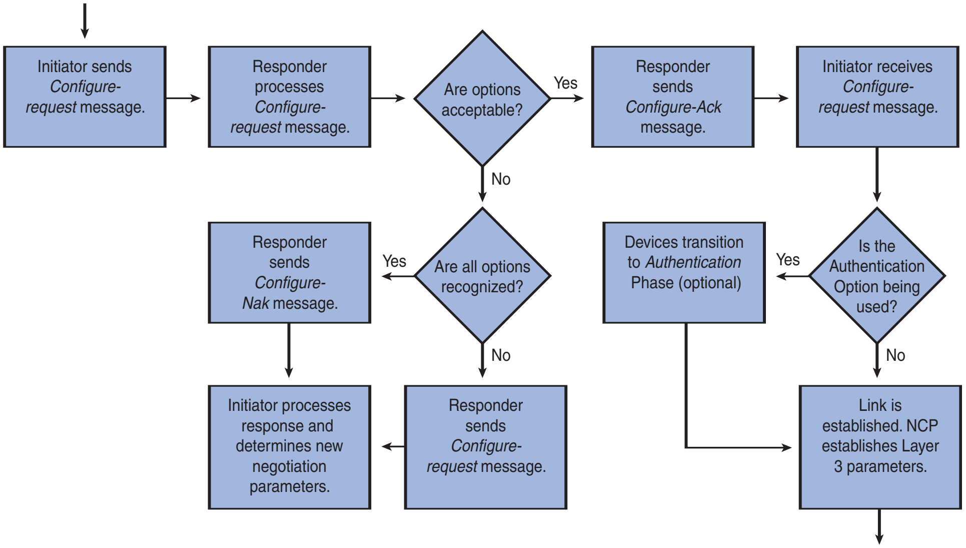
- **Code-Reject and Protocol-Reject**: These frame types provide feedback when one device receives an invalid frame due to either an unrecognized LCP code (LCP frame type) or a bad protocol identifier. For example, if an uninterpretable packet is received from the peer, a Code-Reject packet is sent in response.
- **Echo-Request, Echo-Reply, and Discard-Request**: These frames can be used to test the link.

When the transfer of data at the network layer is complete, the LCP terminates the link. In the figure, notice that the NCP only terminates the network layer and NCP link. The link remains open until the LCP terminates it. If the LCP terminates the link before the NCP, the NCP session is also terminated.

PPP can terminate the link at any time. This might happen because of the loss of the carrier, authentication failure, link quality failure, the expiration of an idle-period timer, or the administrative closing of the link. The LCP closes the link by exchanging Terminate packets. The device initiating the shutdown sends a Terminate-Request message. The other device replies with a Terminate-Ack. A termination request indicates that the device sending it needs to close the link. When the link is closing, PPP informs the network layer protocols so that they may take appropriate action.

Figure 2-29 shows a logical diagram of the LCP link negotiation process.

Figure 2-29 LCP Link Negotiation Process

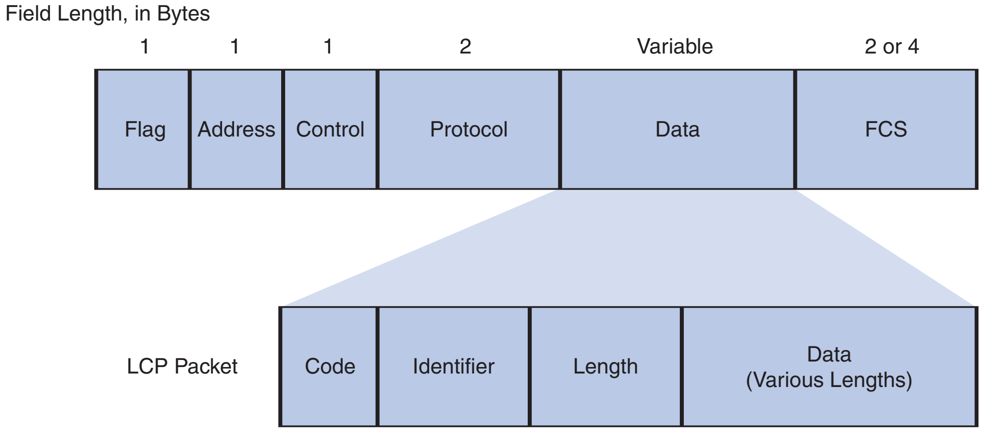

Initiator sends *Configure-request* message.
Responder processes *Configure-request* message.
Are options acceptable?
Yes
Responder sends *Configure-Ack* message.
Initiator receives *Configure-request* message.
No
Responder sends *Configure-Nak* message.
Yes
Are all options recognized?
Devices transition to *Authentication* Phase (optional)
Yes
Is the Authentication Option being used?
No
No
Initiator processes response and determines new negotiation parameters.
Responder sends *Configure-request* message.
Link is established. NCP establishes Layer 3 parameters.

LCP Packet

Each LCP packet is a single LCP message consisting of an LCP Code field identifying the type of LCP packet, an Identifier field so that requests and replies can be matched, and a Length field indicating the size of the LCP packet and LCP packet type-specific data.

Figure 2-30 shows the fields in an LCP packet.

Figure 2-30 LCP Packet Codes



An LCP packet contains the following information:

- **Code**: The Code field is one octet in length and identifies the type of LCP packet.
- **Identifier**: The Identifier field is one octet in length and is used to match packet requests and replies.
- **Length**: The Length field is two octets in length and indicates the total length (including all fields) of the LCP packet.
- **Data**: The Data field is zero or more octets, as indicated by the Length field. The format of this field is determined by the code.

Each LCP packet has a specific function in the exchange of configuration information, depending on its packet type. The Code field of the LCP packet identifies the packet type, as shown in Table 2-4.

Table 2-4 LCP Packet Fields

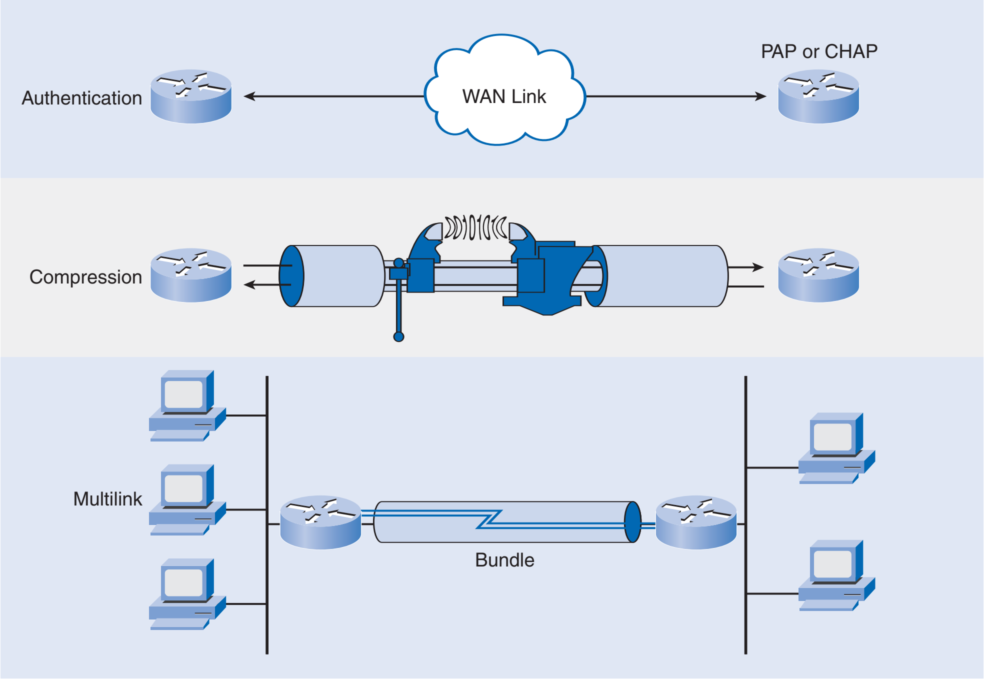
LCP Code	LCP Packet Type	Description
1	Configure-Request	Sent to open or reset a PPP connection. Configure-Request contains a list of LCP options with changes to default option values.
2	Configure-Ack	Sent when all the values of all the LCP options in the last Configure-Request received are recognized and acceptable. When both PPP peers send and receive Configure-Acks, the LCP negotiation is complete.
3	Configure-Nak	Sent when all the LCP options are recognized, but the values of some options are unacceptable. Configure-Nak includes the offending options and their acceptable values.
4	Configure-Reject	Sent when LCP options are unrecognized or unacceptable for negotiation. Configure-Reject includes the unrecognized or nonnegotiable options.
5	Terminate-Request	Optionally sent to close the PPP connection.
6	Terminate-Ack	Sent in response to the Terminate-Request.
7	Code-Reject	Sent when the LCP code is unknown. The Code-Reject message includes the offending LCP packet.
8	Protocol-Reject	Sent when the PPP frame contains an unknown Protocol ID. The Protocol-Reject message includes the offending LCP packet. Protocol-Reject typically is sent by a PPP peer in response to a PPP NCP for a LAN protocol not enabled on the PPP peer.
9	Echo-Request	Optionally sent to test the PPP connection.
10	Echo-Reply	Sent in response to an Echo-Request. The PPP Echo-Request and Echo-Reply are not related to the ICMP Echo Request and Echo Reply messages.
11	Discard-Request	Optionally sent to exercise the link in the outbound direction.

PPP Configuration Options

As shown in Figure 2-31, PPP can be configured to support various functions, including

- Authentication using either PAP or CHAP
- Compression using either Stacker or Predictor
- Multilink, which combines two or more channels to increase the WAN bandwidth

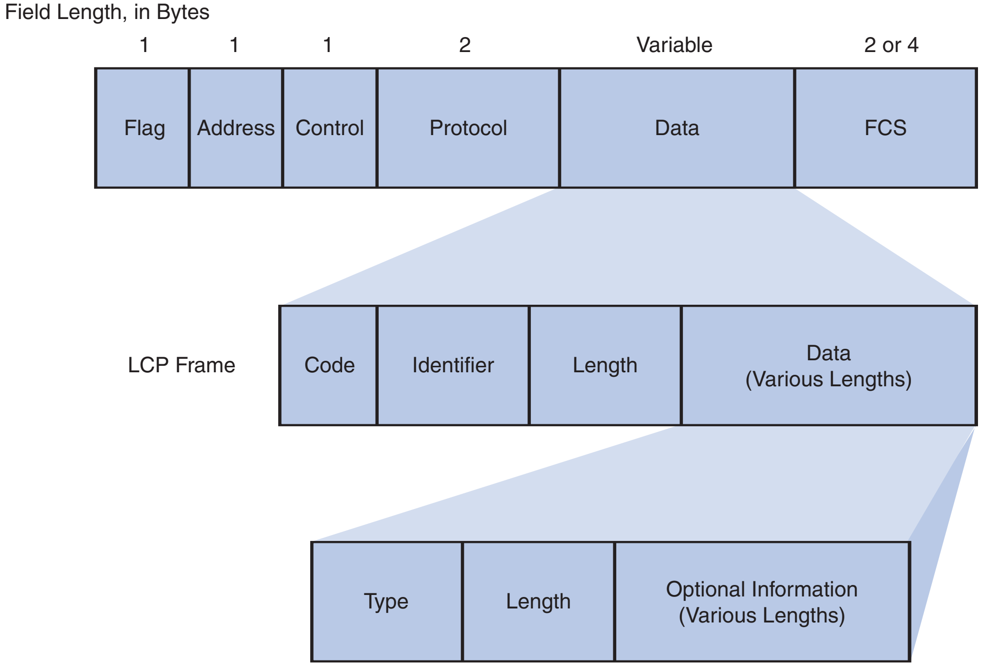
Figure 2-31 PPP Configuration Options



These options are discussed in more detail in the next section.

To negotiate the use of these PPP options, the LCP link-establishment frames contain Option information in the Data field of the LCP frame, as shown in Figure 2-32. If a configuration option is not included in an LCP frame, the default value for that configuration option is assumed.

Figure 2-32 LCP Option Field



This phase is complete when a configuration acknowledgment frame has been sent and received.

NCP Explained

After the link has been initiated, the LCP passes control to the appropriate NCP. Although initially designed for IP datagrams, PPP can carry data from many types of network layer protocols by using a modular approach in its implementation. It can also carry two or more Layer 3 protocols simultaneously. Its modular model allows the LCP to set up the link and then hand the details of a network protocol to a specific NCP. Each network protocol has a corresponding NCP. Each NCP has a corresponding RFC. There are NCPs for IP, IPX, AppleTalk, and others. NCPs use the same packet format as the LCPs.

NCP Process

After the LCP has configured and authenticated the basic link, the appropriate NCP is invoked to complete the specific configuration of the network layer protocol being used. When the NCP has successfully configured the network layer protocol, the network protocol is in the open state on the established LCP link. At this point, PPP can carry the corresponding network layer protocol packets.

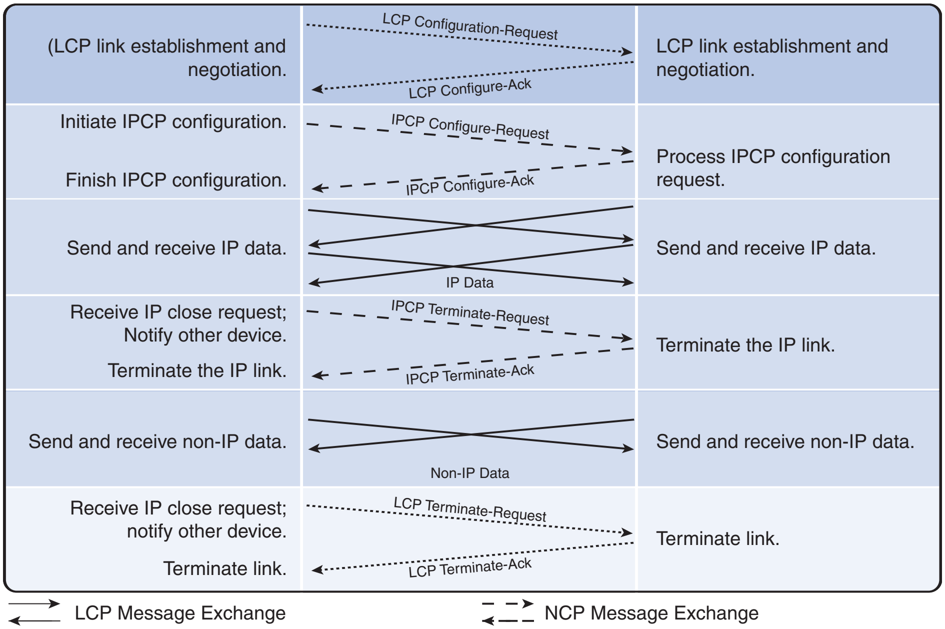
As an example of how the NCP layer works, IP, which is the most common Layer 3 protocol, is used. After LCP has established the link, the routers exchange IPCP messages, negotiating options specific to the protocol. IPCP is responsible for configuring, enabling, and disabling the IP modules on both ends of the link.

IPCP negotiates two options:

- **Compression**: Allows devices to negotiate an algorithm to compress TCP and IP headers and save bandwidth. Van Jacobson TCP/IP header compression reduces the size of the TCP/IP headers to as few as 3 bytes. This can be a significant improvement on slow serial lines, particularly for interactive traffic.
- **IP-Address**: Allows the initiating device to specify an IP address to use for routing IP over the PPP link, or to request an IP address for the responder. Dialup network links commonly use the IP address option.

When the NCP process is complete, the link goes into the open state, and LCP takes over again. Link traffic consists of any possible combination of LCP, NCP, and network layer protocol packets. Figure 2-33 shows how LCP messages can then be used by either device to manage or debug the link.

Figure 2-33 NCP Process


(LCP link establishment and negotiation.
LCP Configuration-Request
LCP Configure-Ack
LCP link establishment and negotiation.
Initiate IPCP configuration.
Finish IPCP configuration.
IPCP Configure-Request
IPCP Configure-Ack
Process IPCP configuration request.
Send and receive IP data.
IP Data
Send and receive IP data.
Receive IP close request; Notify other device.
Terminate the IP link.
IPCP Terminate-Request
IPCP Terminate-Ack
Terminate the IP link.
Send and receive non-IP data.
Non-IP Data
Send and receive non-IP data.
Receive IP close request; notify other device.
Terminate link.
LCP Terminate-Request
LCP Terminate-Ack
Terminate link.
LCP Message Exchange
NCP Message Exchange

PPP Configuration Options

Basic PPP configuration is similar to configuring other Layer 2 protocols such as HDLC. PPP also includes several configuration options, including authentication and compression.

PPP Configuration Options

In the preceding section, you were introduced to the LCP options you can configure to meet specific WAN connection requirements. PPP may include the following LCP options:

- **Authentication**: Peer routers exchange authentication messages. Two authentication choices are Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP). Authentication is explained in the next section.
- **Compression**: Increases the effective throughput on PPP connections by reducing the amount of data in the frame that must travel across the link. The protocol decompresses the frame at its destination. Two compression protocols available in Cisco routers are Stacker and Predictor.
- **Error detection**: Identifies fault conditions. The Quality and Magic Number options help ensure a reliable, loop-free data link. The Magic Number field helps detect links that are in a looped-back condition. Until the Magic-Number Configuration Option has been successfully negotiated, the Magic Number must be transmitted as 0. Magic numbers are generated randomly at each end of the connection.
- **Multilink**: Cisco IOS software Release 11.1 and later support Multilink PPP. This alternative provides load balancing over the router interfaces that PPP uses. Multilink PPP (also called MP, MPPP, MLP, or Multilink) provides a method for spreading traffic across multiple physical WAN links while providing packet ***fragmentation*** and ***reassembly***, proper sequencing, multivendor interoperability, and load balancing on inbound and outbound traffic.
- **PPP Callback**: To enhance security, Cisco IOS software Release 11.1 and later offer callback over PPP. With this LCP option, a Cisco router can act as a callback client or callback server. The client makes the initial call, requests that the server call it back, and terminates its initial call. The callback router answers the initial call and makes the return call to the client based on its configuration statements. The command is **ppp callback** [**accept** | **request**].

When options are configured, a corresponding field value is inserted into the LCP option field. Table 2-5 highlights the valid LCP option values.

Table 2-5 Configurable Option Field Codes

Option Name	Option Type	Option Length	Description
Maximum Receive Unit (MRU)	1	4	The maximum size of a PPP frame. Cannot exceed 65,535 bytes. The default is 1500 bytes. If neither peer is changing the default, it is not negotiated.
Asynchronous Control Character Map (ACCM)	2	6	A bit map that enables character escapes for asynchronous links. By default, character escapes are used.
Authentication Protocol	3	5 or 6	This field indicates the authentication protocol, either PAP or CHAP.
Magic Number	5	6	A random number chosen to distinguish a peer and detect looped-back lines.
Protocol Compression	7	2	A flag indicating that the PPP protocol ID be compressed into a single octet when the 2-byte protocol ID is in the range 0x00-00 to 0x00-FF.
Address and Control Field Compression	8	2	A flag indicating that the PPP Address field (always set to 0xFF) and the PPP Control field (always set to 0x03) should be removed from the PPP header.
Callback	13 or 0x0D	3	A one-octet indicator of how callback is to be determined.

PPP Configuration Commands

This series of examples shows you how to configure PPP and some of the options.

Example 1: Enabling PPP on an Interface

To set PPP as the encapsulation method used by a serial or ISDN interface, use the **encapsulation ppp** interface configuration command.

Example 2-3 enables PPP encapsulation on serial interface 0/0/0.

Example 2-3 Configuring PPP Encapsulation

```
R3# configure terminal
R3(config)# interface serial 0/0/0
R3(config-if)# encapsulation ppp
```

The **encapsulation ppp** command has no arguments. However, you must first configure the router with an IP routing protocol to use PPP encapsulation. You should recall that if you do not configure PPP on a Cisco router, the default encapsulation for serial interfaces is HDLC.

Example 2: Compression

You can configure point-to-point software compression on serial interfaces after you have enabled PPP encapsulation. Because this option invokes a software compression process, it can affect system performance. If the traffic already consists of compressed files (.zip, .tar, or .mpeg, for example), using compression on the router would achieve little benefit. The command syntax for the **compress** command is as follows:

```
Router(config-if)# compress [predictor | stac]
```

where

- **predictor** (optional) specifies that a predictor compression algorithm will be used.
- **stac** (optional) specifies that a Stacker (LZS) compression algorithm will be used.

Example 2-4 configures compression over PPP.

Example 2-4 Configuring Compression

```
R3(config)# interface serial 0/0/0
R3(config-if)# encapsulation ppp
R3(config-if)# compress [predictor | stac]
```

Example 3: Link Quality Monitoring

Recall from our discussion of LCP phases that LCP provides an optional link quality determination phase. In this phase, LCP tests the link to determine whether the link quality is sufficient to use Layer 3 protocols. The following command ensures that the link meets the quality requirement you set; otherwise, the link closes down:

```
Router(config-if)#ppp quality percentage
```

The *percentage* parameter specifies the link quality threshold. The range is 1 to 100.

The percentages are calculated for both incoming and outgoing directions. The outgoing quality is calculated by comparing the total number of packets and bytes sent to the total number of packets and bytes received by the destination node. The incoming quality is calculated by comparing the total number of packets and bytes received to the total number of packets and bytes sent by the destination node.

If the link quality percentage is not maintained, the link is deemed to be of poor quality and is taken down. Link Quality Monitoring (LQM) implements a time lag so that the link does not bounce up and down.

Example 2-5 configuration monitors the data dropped on the link and avoids frame looping.

Example 2-5 Configuring Link Quality Monitoring

```
R3(config)# interface serial 0/0/0
R3(config-if)# encapsulation ppp
R3(config-if)# ppp quality 80
```

Use the **no ppp quality** command to disable LQM.

Example 4: Load Balancing Across Links

Multilink PPP (also called MP, MPPP, MLP, or Multilink) provides a method for spreading traffic across multiple physical WAN links while providing packet fragmentation and reassembly, proper sequencing, multivendor interoperability, and load balancing on inbound and outbound traffic.

MPPP allows packets to be fragmented. It sends these fragments simultaneously over multiple point-to-point links to the same remote address. The multiple physical links come up in response to a user-defined load threshold. MPPP can measure the load on just inbound traffic, or on just outbound traffic, but not on the combined load of both inbound and outbound traffic.

The commands shown in Example 2-6 perform load balancing across multiple links.

Example 2-6 Configuring Load Balancing

```
R3(config)# interface serial 0/0/0
R3(config-if)# encapsulation ppp
R3(config-if)# ppp multilink
```

The **multilink** command has no arguments. To disable PPP multilink, use the **no ppp multilink** command.

Verifying a Serial PPP Encapsulation Configuration

Use the **show interfaces serial** command to verify proper configuration of HDLC or PPP encapsulation. The command output in Example 2-7 shows a PPP configuration.

Example 2-7 Verifying a Serial PPP Encapsulation Configuration

```
R2# show interfaces serial 0/0/0

Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Open: CDPCP, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:07, output 00:00:07, output hang never
  Last clearing of "show interface" counters 00:00:11
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/1/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 96 kilobits/sec
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     6 packets input, 76 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     7 packets output, 84 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

When you configure HDLC, the output of the **show interfaces serial** command should show **Encapsulation HDLC**. When you configure PPP, you can check its LCP and NCP states.

Table 2-6 summarizes commands used when verifying PPP.

Table 2-6 Verifying and Debugging Commands

Command	Description
show interfaces	Displays statistics for all interfaces configured on the router or access server.
show interfaces serial	Displays information about a serial interface.
debug ppp	Debugs PPP.
undebug all	Turns off all debugging displays.

Troubleshooting PPP Encapsulation

By now you are aware that the **debug** command is used for troubleshooting and is accessed from privileged EXEC mode of the command-line interface. **debug** displays information in real time about various router operations and the related traffic the router generates or receives, as well as any error messages. It is a very useful and informative tool, but you must always remember that Cisco IOS treats **debug** as a high-priority task. It can consume a significant amount of resources, and the router is forced to process-switch the packets being debugged. **debug** must not be used as a monitoring tool; it is meant to be used for a short period of time for troubleshooting. When troubleshooting a serial connection, you use the same approach you have used in other configuration tasks.

Troubleshooting the Serial Encapsulation Configuration

Use the **debug ppp** command to display information about the operation of PPP. The command syntax for this command is as follows:

```
debug ppp {packet | negotiation | error | authentication | compression | cbcp}
```

The **no** form of this command disables debugging output.

Table 2-7 explains the command parameters for the **debug ppp** command.

Table 2-7 **debug ppp** Command Parameters

Parameter	Usage
packet	Displays PPP packets being sent and received. (This command displays low-level packet dumps.)
negotiation	Displays PPP packets transmitted during PPP startup, where PPP options are negotiated.

Table 2-7 **debug ppp** Command Parameters

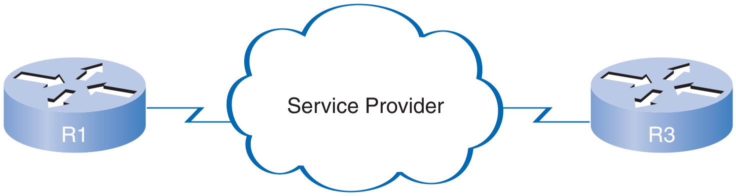
Parameter	Usage
error	Displays protocol errors and error statistics associated with PPP connection negotiation and operation.
authentication	Displays authentication protocol messages, including Challenge Handshake Authentication Protocol (CHAP) packet exchanges and Password Authentication Protocol (PAP) exchanges.
compression	Displays information specific to the exchange of PPP connections using Microsoft Point-to-Point Compression (MPPC). This command is useful for obtaining incorrect packet sequence number information where MPPC compression is enabled.
cbcp	Displays protocol errors and statistics associated with PPP connection negotiations using Microsoft Callback Control Protocol (MSCB).

Output of the **debug ppp packet** Command

A good command to use when troubleshooting serial interface encapsulation is **debug ppp packet**.

The topology shown in Figure 2-34 will be used for the next series of output examples.

Figure 2-34 Sample Topology



Example 2-8 is output from the **debug ppp packet** command as seen from the Link Quality Monitor (LQM) side of the connection. The output shows the packet exchange between router R1 and router R3 during normal PPP operation.

Example 2-8 Output of the **debug ppp packet** Command

```
R3# debug ppp packet

PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial2(i): pkt type 0xC025, datagramsize 52
PPP Serial2(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial2(i): pkt type 0xC021, datagramsize 16
PPP Serial2: I LCP ECHOREQ(9) id 3  magic D3454
PPP Serial2: input(C021) state = OPEN code = ECHOREQ(9) id = 3 len = 12
PPP Serial2: O LCP ECHOREP(A) id 3  magic D21B4
PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial2(i): pkt type 0xC025, datagramsize 52
PPP Serial2(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial2(i): pkt type 0xC021, datagramsize 16
PPP Serial2: I LCP ECHOREQ(9) id 4  magic D3454
PPP Serial2: input(C021) state = OPEN code = ECHOREQ(9) id = 4 len = 12
PPP Serial2: O LCP ECHOREP(A) id 4  magic D21B4
PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial2(i): pkt type 0xC025, datagramsize 52
PPP Serial2(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial2(i): pkt type 0xC021, datagramsize 16
PPP Serial2: I LCP ECHOREQ(9) id 5  magic D3454
PPP Serial2: input(C021) state = OPEN code = ECHOREQ(9) id = 5 len = 12
PPP Serial2: O LCP ECHOREP(A) id 5  magic D21B4
PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial2(i): pkt type 0xC025, datagramsize 52
PPP Serial2(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial2(i): pkt type 0xC021, datagramsize 16
PPP Serial2: I LCP ECHOREQ(9) id 6  magic D3454
PPP Serial2: input(C021) state = OPEN code = ECHOREQ(9) id = 6 len = 12
PPP Serial2: O LCP ECHOREP(A) id 6  magic D21B4
PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial2(i): pkt type 0xC025, datagramsize 52
PPP Serial2(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial2(i): pkt type 0xC021, datagramsize 16
PPP Serial2: I LCP ECHOREQ(9) id 7  magic D3454
PPP Serial2: input(C021) state = OPEN code = ECHOREQ(9) id = 7 len = 12
PPP Serial2: O LCP ECHOREP(A) id 7  magic D21B4
PPP Serial2(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
```

This display example depicts packet exchanges under normal PPP operation. This is only a partial listing, but it’s enough to get you ready for the practice lab.

Look at each line in the output, and match it to the meaning of the field. Although it's beyond the scope of this course, use the following to guide your examination of the output.

- **PPP**: PPP debugging output.
- **Serial2**: The interface number associated with this debugging information.
- **(o), O**: The detected packet is an output packet.
- **(i), I**: The detected packet is an input packet.
- **lcp_slqr()**: Procedure name; running LQM, send a Link Quality Report (LQR).
- **lcp_rlqr()**: Procedure name; running LQM, received an LQR.
- **input (C021)**: Router received a packet of the specified packet type (in hexadecimal). A value of C025 indicates a packet of type LQM.
- **state = OPEN**: PPP state; normal state is OPEN.
- **magic = D21B4**: Magic Number for indicated node. When output is indicated, this is the Magic Number of the node on which debugging is enabled. The actual Magic Number depends on whether the packet detected is indicated as I or O.
- **datagramsize = 52**: Packet length, including header.
- **code = ECHOREQ(9)**: Identifies the type of packet received in both string and hexadecimal form.
- **len = 48**: Packet length without header.
- **id = 3**: ID number per Link Control Protocol (LCP) packet format.
- **pkt type 0xC025**: Packet type in hexadecimal. Typical packet types are C025 for LQM and C021 for LCP.
- **LCP ECHOREQ (9)**: Echo Request. Value in parentheses is the hexadecimal representation of the LCP type.
- **LCP ECHOREP (A)**: Echo Reply. Value in parentheses is the hexadecimal representation of the LCP type.

Output of the **debug ppp negotiation** Command

Example 2-9 shows the output of the **debug ppp negotiation** command in a normal negotiation, where both sides agree on network control program (NCP) parameters.

Example 2-9 Output of the **debug ppp negotiation** Command

```
R1# debug ppp negotiation

ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
ppp: received config for type = 4 (QUALITYTYPE) acked
ppp: received config for type = 5 (MAGICNUMBER) value = 3D567F8 acked (ok)
PPP Serial2: state = ACKSENT fsm_rconfack(C021): rcvd id 5
ppp: config ACK received, type = 4 (CI_QUALITYTYPE), value = C025
ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
ppp: ipcp_reqci: returning CONFACK. (ok)
PPP Serial2: state = ACKSENT fsm_rconfack(8021): rcvd id 4
```

In this case, protocol type IP is proposed and acknowledged. Let's take the output a line or two at a time.

The first two lines indicate that the router is trying to bring up the LCP and will use the indicated negotiation options (Quality Protocol and Magic Number):

```
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
```

The value fields are the values of the options themselves. C025/3E8 translates to Quality Protocol LQM. 3E8 is the reporting period (in hundredths of a second). 3D56CAC is the value of the Magic Number for the router.

The next two lines indicate that the other side negotiated for options 4 and 5 and that it requested and acknowledged both:

```
ppp: received config for type = 4 (QUALITYTYPE) acked
ppp: received config for type = 5 (MAGICNUMBER) value = 3D567F8 acked (ok)
```

If the responding end does not support the options, the responding node sends a CONFREJ. If the responding end does not accept the value of the option, it sends a CONFNAK with the value field modified.

The next three lines indicate that the router received a CONFACK from the responding side and displays accepted option values:

```
PPP Serial4: state = ACKSENT fsm_rconfack(C021): rcvd id 5
ppp: config ACK received, type = 4 (CI_QUALITYTYPE), value = C025
ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
```

Use the rcvd id field to verify that the CONFREQ and CONFACK have the same ID field.

The next line indicates that the router has IP routing enabled on this interface and that the IPCP NCP negotiated successfully:

```
ppp: ipcp_reqci: returning CONFACK
(ok)
```

Output of the **debug ppp error** Command

You can use the **debug ppp error** command to display protocol errors and error statistics associated with PPP connection negotiation and operation, as shown in Example 2-10. These messages might appear when the Quality Protocol option is enabled on an interface that is already running PPP.

Example 2-10 Output of the **debug ppp error** Command

```
R1# debug ppp error

PPP Serial3(i): rlqr receive failure. successes = 15
PPP: myrcvdiffp = 159 peerxmitdiffp = 41091
PPP: myrcvdiffo = 2183 peerxmitdiffo = 1714439
PPP: threshold = 25
PPP Serial2(i): rlqr transmit failure. successes = 15
PPP: myxmitdiffp = 41091 peerrcvdiffp = 159
PPP: myxmitdiffo = 1714439 peerrcvdiffo = 2183
PPP: l->OutLQRs = 1 LastOutLQRs = 1
PPP: threshold = 25
PPP Serial3(i): lqr_protrej() Stop sending LQRs.
PPP Serial3(i): The link appears to be looped back.
```

Look at each line in the output, and match it to the meaning of the field. Again, this is beyond the scope of this course, but you can use the following to guide your examination of the output.

- **PPP**: PPP debugging output.
- **Serial3(i)**: The interface number associated with this debugging information. Indicates that this is an input packet.
- **rlqr receive failure**: The receiver does not accept the request to negotiate the Quality Protocol option.
- **myrcvdiffp = 159**: The number of packets received over the time period specified.
- **peerxmitdiffp = 41091**: The number of packets sent by the remote node over this period.
- **myrcvdiffo = 2183**: The number of octets received over this period.
- **peerxmitdiffo = 1714439**: The number of octets sent by the remote node over this period.
- **threshold = 25**: The maximum error percentage acceptable on this interface. You calculate this percentage using a value of 100 minus the threshold value entered in the **ppp quality** *percentage* interface configuration command. In this case, the maximum error percentage calculated is set to 25% which means that the interface was configured using the **ppp quality 75** command (100 – 75 = 25). This means that the local router

must maintain a minimum 75 percent nonerror percentage (or a 25% maximum error percentage), or the PPP link closes down.

- **OutLQRs = 1**: The current send LQR sequence number of the local router.
- **LastOutLQRs = 1**: The last sequence number that the remote node side has seen from the local node.

Packet Tracer Activity

Configuring Point-to-Point Encapsulations 2.3.4

In this activity, you will practice changing the encapsulation on serial interfaces. Detailed instructions are provided within the activity. Use File e4-234.pka on the CD-ROM that accompanies this book to perform this activity using Packet Tracer.

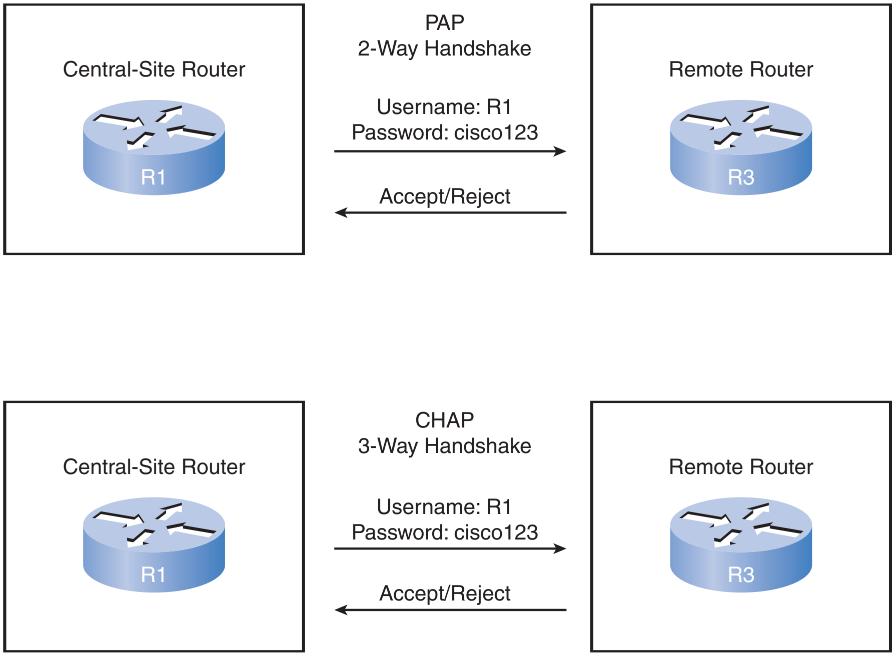
PPP Authentication Protocols

PPP defines an extensible version of LCP that allows negotiation of an authentication protocol for a router to authenticate its peer before allowing network layer protocols to transmit over the link.

PPP Authentication Protocol

RFC 1334 defines two protocols for authentication, as shown in Figure 2-35.

Figure 2-35 PPP Authentication Protocols



Password Authentication Protocol (PAP) is a basic two-way process. There is no encryption, because the username and password are sent in plain text. If it is accepted, the connection is allowed. Challenge Handshake Authentication Protocol (CHAP) is more secure than PAP. It involves a three-way exchange of a shared secret. The process is described later in this section.

The authentication phase of a PPP session is optional. If it is used, you can authenticate the peer after the LCP establishes the link and choose the authentication protocol. If it is used, authentication takes place before the network layer protocol configuration phase begins.

The authentication options require that the calling side of the link enter authentication information. This helps ensure that the user has permission from the network administrator to make the call. Peer routers exchange authentication messages.

Password Authentication Protocol

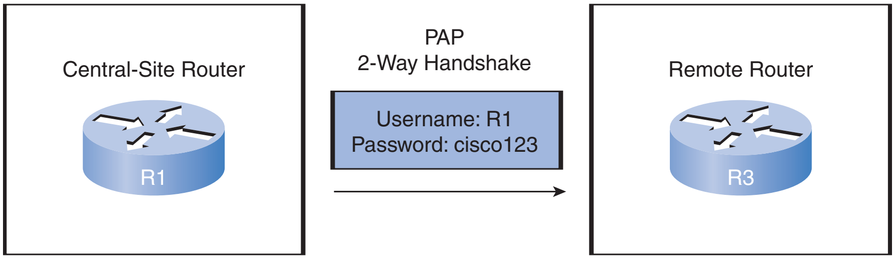
One of the many features of PPP is that it performs Layer 2 authentication in addition to other layers of authentication, encryption, access control, and general security procedures.

Initiating PAP

PAP provides a simple method for a remote node to establish its identity using a two-way handshake. PAP is not interactive. When the **ppp authentication pap** command is used, the username and password are sent as one LCP data package, rather than the server's sending a login prompt and waiting for a response.

Figure 2-36 shows that after PPP completes the link establishment phase, the remote node repeatedly sends a username-password pair across the link until the sending node acknowledges it or terminates the connection.

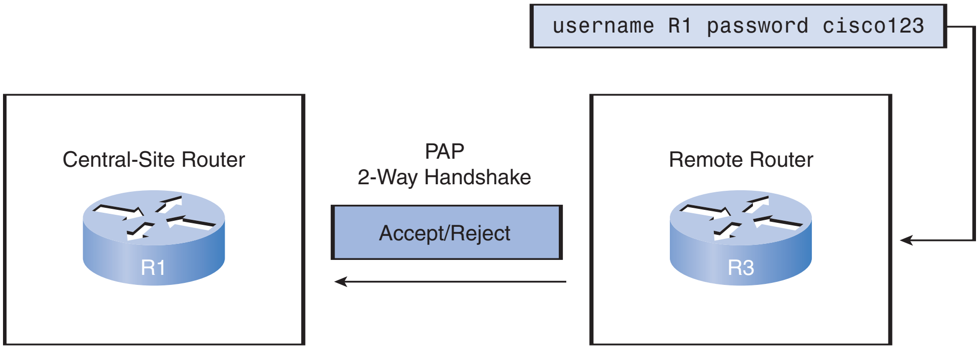
Figure 2-36 Initiating PAP



Router R1 sends its PAP username and password to Router R3.

Figure 2-37 shows that at the receiving node, the router (or an authentication server) checks the username-password. It either allows or denies the connection. An accept or reject message is returned to the requester.

Figure 2-37 Completing PAP



```
username R1 password cisco123
```

Router R3 evaluates R1’s username and password against its local database. If it matches, it accepts the connection. If not, it rejects the connection.

PAP is not a strong authentication protocol. Using PAP, you send passwords across the link in clear text, and there is no protection from playback or repeated trial-and-error attacks. The remote node is in control of the frequency and timing of the login attempts.

Nonetheless, sometimes using PAP can be justified. For example, despite its shortcomings, PAP may be used in the following environments:

- A large installed base of client applications that do not support CHAP
- Incompatibilities between different vendor implementations of CHAP
- Situations in which a plain-text password must be available to simulate a login at the remote host

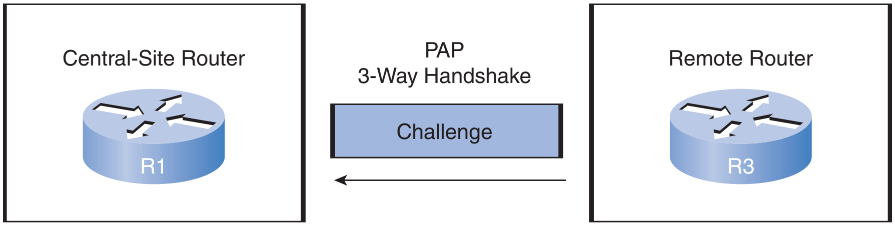
Challenge Handshake Authentication Protocol (CHAP)

Challenge Handshake Authentication Protocol (CHAP), defined in RFC 1994, verifies the peer’s identity by means of a three-way handshake. CHAP is considered a stronger authentication method than PAP.

PAP authenticates only once. After authentication is established, PAP essentially stops working, because it can’t reauthenticate during the session. This leaves the network vulnerable to attack. Unlike PAP, which authenticates only once, CHAP conducts periodic challenges to make sure that the remote node still has a valid password value. The password value is variable and changes unpredictably while the link exists.

After the PPP link establishment phase is complete, the local router sends a challenge message to the remote node, as shown in Figure 2-38.

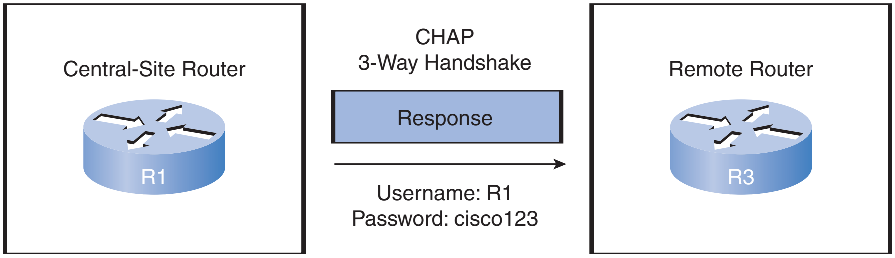
Figure 2-38 Initiating CHAP



Router R3 initiates the 3-way handshake and sends a challenge message to Router R1.

Figure 2-39 shows that the remote node responds with a value calculated using a one-way hash function, which typically is ***Message Digest 5 (MD5)***, based on the password and challenge message. These steps are explained in the next section.

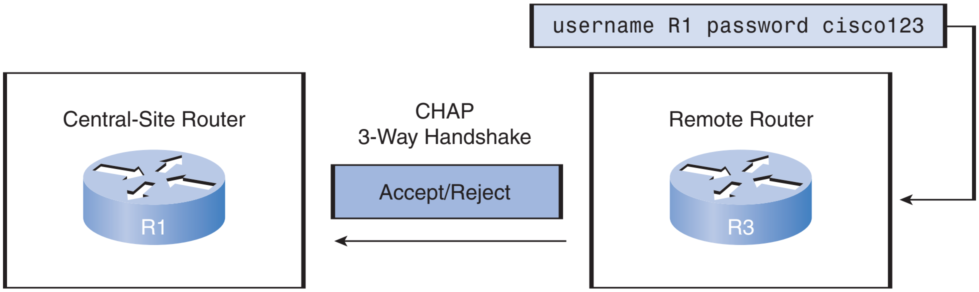
Figure 2-39 Responding to CHAP



R1 responds R3's CHAP challenge by sending its CHAP username and password.

In Figure 2-40, the local router checks the response against its own calculation of the expected hash value. If the values match, the initiating node acknowledges the authentication. Otherwise, it immediately terminates the connection.

Figure 2-40 Completing CHAP



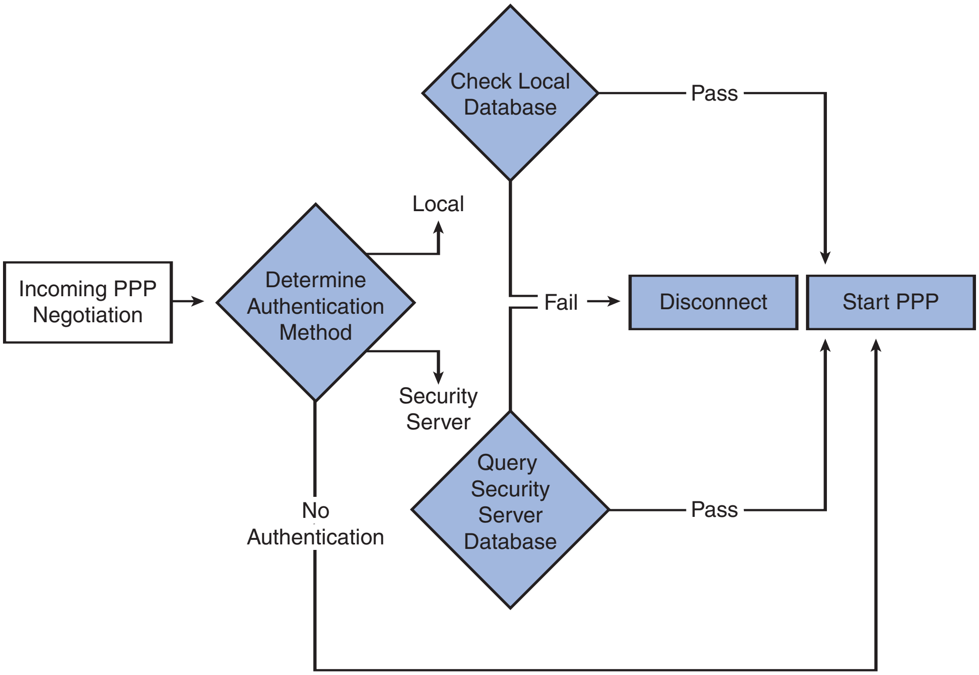
Router R3 evaluates R1's username and password against its local database. If it matches, it accepts the connection. If not, it rejects the connection.

CHAP provides protection against playback attack by using a variable challenge value that is unique and unpredictable. Because the challenge is unique and random, the resulting hash value is also unique and random. The use of repeated challenges limits the time of exposure to any single attack. The local router or a third-party authentication server is in control of the frequency and timing of the challenges.

PPP Encapsulation and Authentication Process

You can use the flowchart shown in Figure 2-41 to help understand the PPP authentication process when configuring PPP. The flowchart provides a visual example of the logic decisions that PPP makes.

Figure 2-41 PPP Encapsulation and Authentication Process

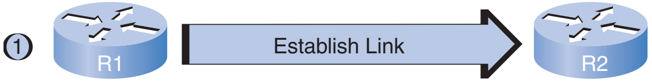


For example, if an incoming PPP request requires no authentication, PPP progresses to the next level. If an incoming PPP request requires authentication, it can be authenticated using either the local database or a security server. As illustrated in the flowchart, successful authentication progresses to the next level, whereas an authentication failure disconnects and drops the incoming PPP request.

Follow these steps to see how router R1 establishes an authenticated PPP CHAP connection with Router R2:

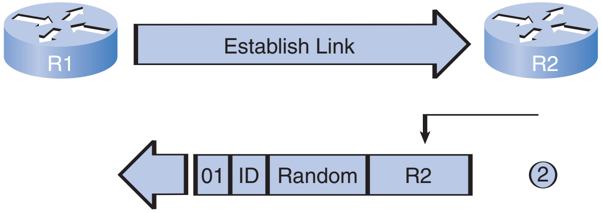
Step 1. As shown in Figure 2-42, router R1 initially negotiates the link connection using LCP with router R2, and the two systems agree to use CHAP authentication during the PPP LCP negotiation.

Figure 2-42 Establishing a Link



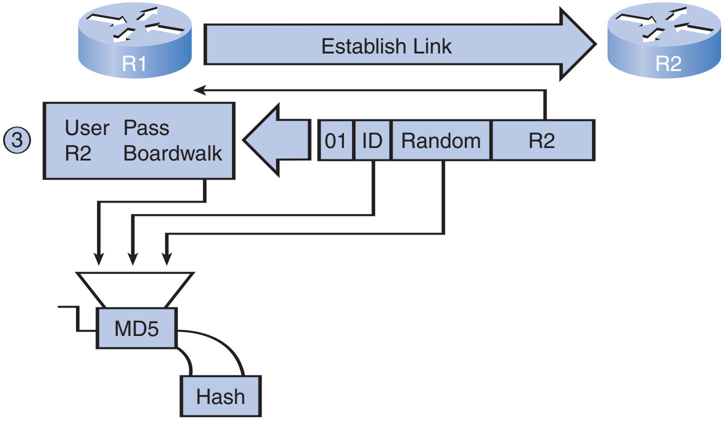
Step 2. As shown in Figure 2-43, router R2 generates an ID and a random number and sends that plus its username as a CHAP challenge packet to R1.

Figure 2-43 Sending a CHAP Challenge to R1



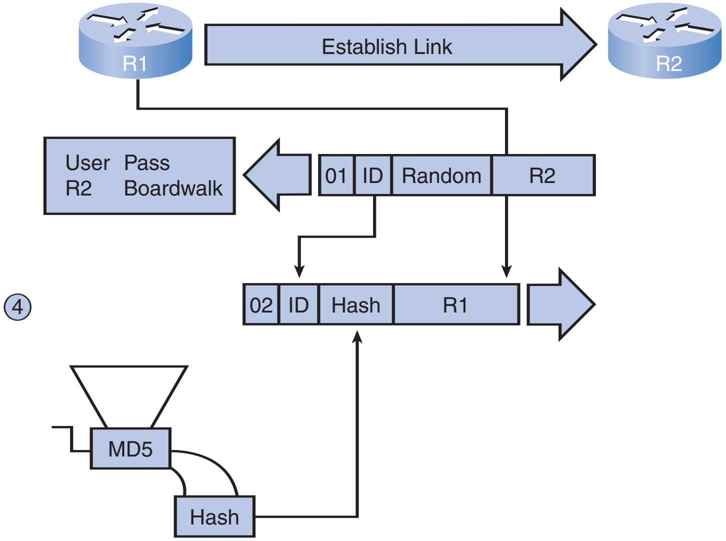
Step 3. As shown in Figure 2-44, router R1 uses the username of the challenger (R2) and cross-references it with its local database to find its associated password. R1 then generates a unique MD5 hash number using R2's username, ID, random number, and the shared secret password.

Figure 2-44 R1 Validates R2



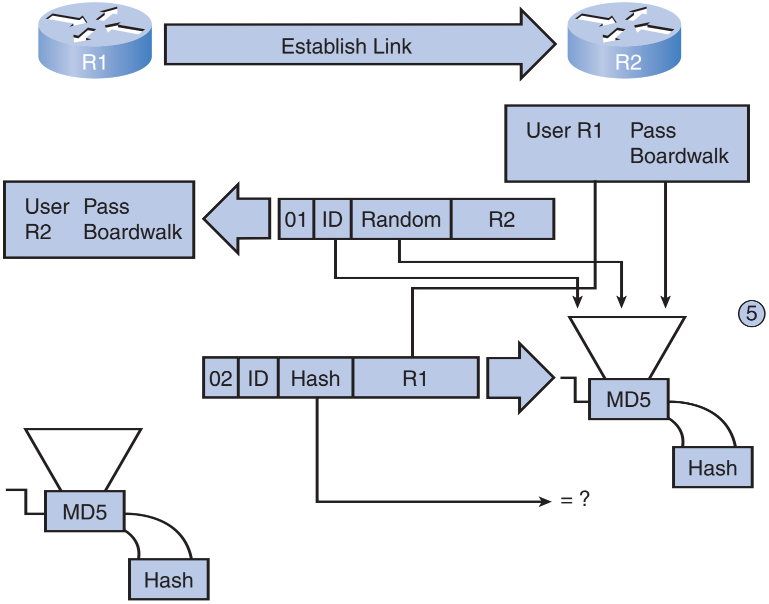
Step 4. As shown in Figure 2-45, router R1 validates router R2's hash and then sends the challenge ID, the hashed value, and its username (R1) to R2.

Figure 2-45 R1 Sends the Challenge to R2



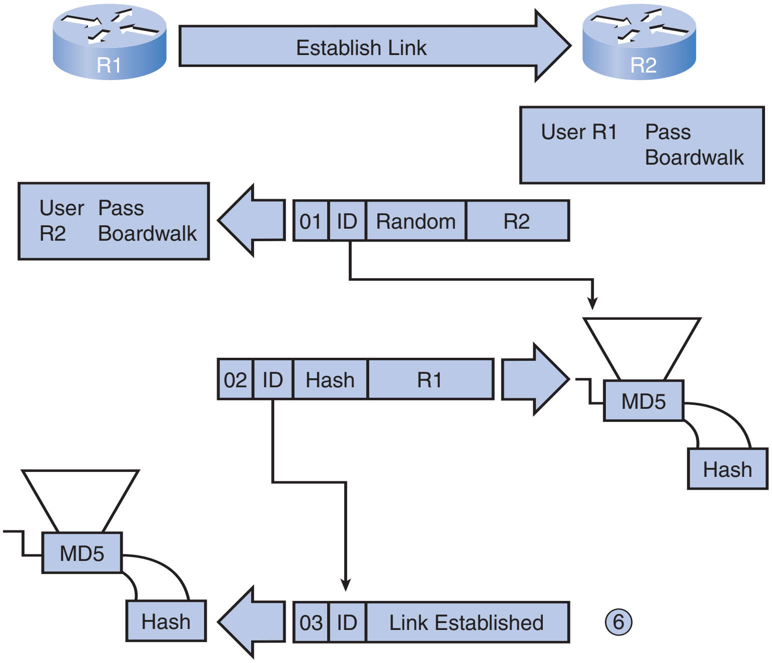
Step 5. As shown in Figure 2-46, router R2 generates its own hash value using the ID, the shared secret password, and the random number it originally sent to R1.

Figure 2-46 R2 Validates R1



Step 6. As shown in Figure 2-47, router R2 compares its hash value with the hash value sent by R1. If the values are the same, R2 sends a link established response to R1.

Figure 2-47 R2 Establishes the Link



If the authentication fails, a CHAP failure packet is built from the following components:

- 04 = CHAP failure message type
- id = copied from the response packet
- "Authentication failure" or a similar message, which is meant to be a user-readable explanation

Note that the shared secret password must be identical on R1 and R2.

Configuring PPP with Authentication

To specify the order in which the CHAP or PAP protocols are requested on the interface, use the **ppp authentication** interface configuration command, the syntax for which is as follows:

```
ppp authentication {chap | chap pap | pap chap | pap} [if-needed]
  [list-name | default [callin]
```

Use the **no** form of the command to disable this authentication.

Table 2-8 describes the **ppp authentication** command parameters.

Table 2-8 ppp authentication Command Parameters

Parameter	Usage
chap	Enables CHAP on a serial interface.
chap pap	Enables both CHAP and PAP and performs CHAP authentication before PAP.
pap chap	Enables both CHAP and PAP and performs PAP authentication before CHAP.
pap	Enables PAP on a serial interface.
if-needed	(Optional) Used with TACACS and XTACACS. Do not perform CHAP or PAP authentication if the user has already provided authentication. This option is available only on asynchronous interfaces.
list-name	(Optional) Used with AAA/TACACS+. Specifies the name of a list of TACACS+ authentication methods to use. If no list name is specified, the system uses the default. Lists are created with the **aaa authentication ppp** command.
default	(Optional) Used with AAA/TACACS+. Created with the **aaa authentication ppp** command.
callin	Specifies authentication on incoming (received) calls only.

After you have enabled CHAP or PAP authentication, or both, the local router requires the remote device to prove its identity before allowing data traffic to flow. This is done as follows:

- PAP authentication requires that the remote device send a name and password to be checked against a matching entry in the local username database or in the remote TACACS/TACACS+ database.
- CHAP authentication sends a challenge to the remote device. The remote device must encrypt the challenge value with a shared secret and return the encrypted value and its name to the local router in a response message. The local router uses the name of the remote device to look up the appropriate secret in the local username or remote ***TACACS/TACACS+*** database. It uses the looked-up secret to encrypt the original challenge and verify that the encrypted values match.

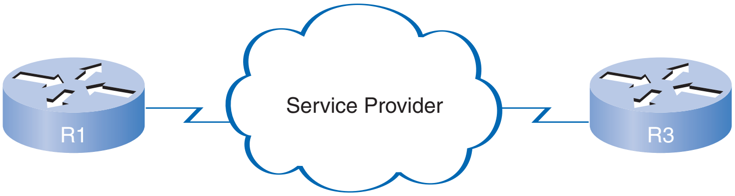
Note

AAA/TACACS is a dedicated server used to authenticate users. AAA stands for authentication, authorization, and accounting. TACACS clients send a query to a TACACS authentication server. The server can authenticate the user, authorize what the user can do, and track what the user has done.

You may enable PAP or CHAP or both. If both methods are enabled, the first method specified is requested during link negotiation. If the peer suggests using the second method or simply refuses the first method, the second method is tried. Some remote devices support CHAP only and some PAP only. The order in which you specify the methods is based on your concerns about the ability of the remote device to correctly negotiate the appropriate method as well as your concern about data line security. PAP usernames and passwords are sent as clear-text strings and can be intercepted and reused. CHAP has eliminated most of the known security holes.

Figure 2-48 shows a sample topology.

Figure 2-48 Sample Topology



Examples 2-11 and 2-12 show a two-way PAP authentication configuration.

Example 2-11 Sample PAP Configuration on R1

```
hostname R1
username R3 password sameone
!
int serial 0/0
ip address 128.0.1.1 255.255.255.0
encapsulation ppp
ppp authentication PAP
ppp pap sent-username R1 password sameone
```

Example 2-12 Sample PAP Configuration on R3

```
hostname R3
username R1 password sameone
!
int serial 0/0
ip address 128.0.1.2 255.255.255.0
encapsulation ppp
ppp authentication PAP
ppp pap sent-username R3 password sameone
```

Both routers authenticate and are authenticated, so the PAP authentication commands mirror each other. The PAP username and password that each router sends must match those specified with the **username** *name* **password** *password* command of the other router.

PAP provides a simple method for a remote node to establish its identity using a two-way handshake. This is done only on initial link establishment. The hostname on one router must match the username the other router has configured. The passwords must also match and are case-sensitive.

CHAP periodically verifies the identity of the remote node using a three-way handshake. The hostname on one router must match the username the other router has configured. The passwords must also match. This occurs on initial link establishment and can be repeated any time after the link has been established. Examples 2-13 and 2-14 show a CHAP configuration.

Example 2-13 Sample CHAP Configuration on R1

```
hostname R1
username R3 password sameone
!
int serial 0/0
ip address 128.0.1.1 255.255.255.0
encapsulation ppp
ppp authentication CHAP
```

Example 2-14 Sample CHAP Configuration on R3

```
hostname R3
username R1 password sameone
!
int serial 0/0
ip address 128.0.1.2 255.255.255.0
encapsulation ppp
ppp authentication CHAP
```

Troubleshooting a PPP Configuration with Authentication

Authentication is a feature that needs to be implemented correctly, or the security of your serial connection may be compromised. Always verify your configuration with the **show interfaces serial** command, in the same way as you did without authentication.

Never assume that your authentication configuration works without testing it. Debugging allows you to confirm your configuration and correct any deficiencies. The command for debugging PPP authentication is **debug ppp authentication**.

Example 2-15 shows sample output for the **debug ppp authentication** command.

Example 2-15 Troubleshooting a PPP Configuration with Authentication

```
R1# debug ppp authentication

Serial0: Unable to authenticate. No name received from peer
Serial0: Unable to validate CHAP response. USERNAME R3 not found.
Serial0: Unable to validate CHAP response. No password defined for USERNAME R3
Serial0: Failed CHAP authentication with remote.
Remote message is Unknown name
Serial0: remote passed CHAP authentication.
Serial0: Passed CHAP authentication with remote.
Serial0: CHAP input code = 4 id = 3 len = 48
```

The following is an interpretation of the output:

Line 1 says that the router is unable to authenticate on interface Serial0 because the peer did not send a name.

Line 2 says the router was unable to validate the CHAP response because USERNAME R3 was not found.

Line 3 says that no password was found for R3. Other possible responses at this line might have been no name received to authenticate, unknown name, no secret for given name, short MD5 response received, or MD5 compare failed.

In the last line, **code = 4** means a failure has occurred. Other code values are as follows:

- 1 = Challenge
- 2 = Response
- 3 = Success
- 4 = Failure

id = 3 is the ID number per LCP packet format.

len = 48 is the packet length without the header.

Packet Tracer
Activity

Configuring PAP and CHAP Authentication 2.4.6

PPP encapsulation allows two different types of authentication: Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP). PAP uses a clear-text password, whereas CHAP invokes a one-way hash that provides more security than PAP. In this activity, you configure both PAP and CHAP as well as review OSPF routing configuration. Detailed instructions are provided within the activity. Use File e4-246.pka on the CD-ROM that accompanies this book to perform this activity using Packet Tracer.

Summary

On completing this chapter, you can describe in conceptual and practical terms why serial point-to-point communications are used to connect your LAN to your service provider WAN, rather than using a connection where communications bits are sent in parallel, which might intuitively seem faster. You can explain how multiplexing allows efficient communications and maximizes the amount of data that can be passed over a communications link. You learned the functions of key components and protocols of serial communications, and you can configure a serial interface with HDLC encapsulation on a Cisco router.

This chapter provided a good basis for comprehending PPP, including its features, components, and architectures. You can explain how a PPP session is established using the functions of the LCP and NCPs. You learned the syntax of the configuration commands and learned about the use of the various options required to configure a PPP connection. You also learned how to use PAP or CHAP authentication protocols to ensure a secure connection. The steps required for verifying and troubleshooting PPP were described. You are now ready to confirm your knowledge in the lab, where you will configure your router to use PPP to connect to a WAN.

Labs

The activities and labs available in the companion *Accessing the WAN, CCNA Exploration Labs and Study Guide* (ISBN 1-58713-212-5) provide hands-on practice with the following topics introduced in this chapter:

Lab 2-1: Basic PPP Configuration (2.5.1)

In this lab, you learn how to configure PPP encapsulation on serial links using the network shown in the topology diagram. You also learn how to restore serial links to their default HDLC encapsulation. Pay special attention to the router's output when you intentionally break PPP encapsulation. This will assist you in the Troubleshooting lab associated with this chapter. Finally, you configure PPP PAP authentication and PPP CHAP authentication.

Lab 2-2: Challenge PPP Configuration (2.5.2)

In this lab, you learn how to configure PPP encapsulation on serial links using the network shown in the topology diagram. You also configure PPP CHAP authentication. If you need assistance, refer to the Basic PPP Configuration lab, but try to do as much on your own as possible.

Lab 2-3: Troubleshooting PPP Configuration (2.5.3)

The routers at your company were configured by an inexperienced network engineer. Several errors in the configuration have resulted in connectivity issues. Your boss has asked you to troubleshoot and correct the configuration errors and document your work. Using your knowledge of PPP and standard testing methods, find and correct the errors. Make sure that all the serial links use PPP CHAP authentication and that all the networks can be reached.

Packet Tracer
Companion

Many of the Hands-on Labs include Packet Tracer Companion Activities, where you can use Packet Tracer to complete a simulation of the lab. Look for this icon in *Accessing the WAN, CCNA Exploration Labs and Study Guide* for Hands-on Labs that have a Packet Tracer Companion.

Check Your Understanding

Complete all the review questions listed here to test your understanding of the topics and concepts in this chapter. Answers are listed in Appendix, "Check Your Understanding and Challenge Questions Answer Key."

1. Match each PPP establishment step with its appropriate sequence number:

   Step 1

   Step 2

   Step 3

   Step 4

   Step 5

   A. Test link quality (optional).

   B. Negotiate Layer 3 protocol options.

   C. Send link-establishment frames to negotiate options such as MTU size, compression, and authentication.

   D. Send configuration-acknowledgment frames.

   E. NCP reaches Open state.

2. Which output from the **show interfaces s0/0/0** command indicates that the far end of a point-to-point link has a different encapsulation set than the local router?

   A. serial 0/0/0 is down, line protocol is down

   B. serial 0/0/0 is up, line protocol is down

   C. serial 0/0/0 is up, line protocol is up (looped)

   D. serial 0/0/0 is up, line protocol is down (disabled)

   E. serial 0/0/0 is administratively down, line protocol is down

3. What is the default encapsulation for serial interfaces on a Cisco router?

   A. HDLC

   B. PPP

   C. Frame Relay

   D. X.25

4. What is the function of the Protocol field in a PPP frame?

   A. It identifies the application layer protocol that will process the frame.

   B. It identifies the transport layer protocol that will process the frame.

   C. It identifies the data link layer protocol encapsulation in the frame's Date field.

   D. It identifies the network layer protocol encapsulated in the frame's Data field.

5. Match each description with its corresponding term:

   Error control

   Authentication protocols

   Allows load balancing

   Compression protocols

   A. Stacker/predictor
   B. Magic number
   C. Multilink
   D. CHAP/PAP
   E. Call in

6. Which of the following statements describe the function of statistical time-division multiplexing (STDM)? (Choose three.)

   A. Multiple data streams share one common channel.
   B. Bit interleaving controls the timing mechanism that places data on the channel.
   C. Time slots are used on a first-come, first-served basis.
   D. STDM was developed to overcome the inefficiency caused by time slots still being allocated even when the channel has no data to transmit.
   E. Sources of data alternate during transmission and are reconstructed at the receiving end.
   F. Priority can be dedicated to one data source.

7. Which of the following describes the serial connection between two routers using the High-level Data Link Control (HDLC) protocol?

   A. Synchronous or asynchronous bit-oriented transmissions using a universal frame format
   B. Synchronous bit-oriented transmissions using a frame format that allows flow control and error detection
   C. Asynchronous bit-oriented transmissions using a frame format derived from the Synchronous Data Link Control (SDLC) protocol
   D. Asynchronous bit-oriented transmissions using a V.35 DTE/DCE interface

8. If an authentication protocol is configured for PPP operation, when is the client or user workstation authenticated?

   A. Before link establishment

   B. During the link establishment phase

   C. Before the network layer protocol configuration begins

   D. After the network layer protocol configuration has ended

9. Why are Network Control Protocols used in PPP?

   A. To establish and terminate data links

   B. To provide authentication capabilities to PPP

   C. To manage network congestion and to allow quality testing of the link

   D. To allow multiple Layer 3 protocols to operate over the same physical link

10. Which statement describes the PAP authentication protocol?

    A. It sends encrypted passwords by default.

    B. It uses a two-way handshake to establish identity.

    C. It protects against repeated trial-and-error attacks.

    D. It requires the same username to be configured on every router.

11. A technician testing the functionality of a recently installed router is unable to ping the serial interface of a remote router. The technician executes the **show interfaces serial 0/0/0** command on the local router and sees the following line in the router:

    Serial0/0/0 is down, line protocol is down

    What are two possible causes of this command output?

    A. The **clock rate** command is missing.

    B. The carrier detect signal is not sensed.

    C. Keepalives are not being sent.

    D. The interface is disabled due to a high error rate.

    E. The interface is shut down.

    F. The cabling is faulty or incorrect.

12. The network administrator is configuring Router1 to connect to Router2 using three-way handshake authentication. Match each description with the command necessary to configure Router1:

Configure the username and password

Enter interface configuration mode

Specify the encapsulation type

Configure authentication

A. **username Router2 password cisco**

B. **username Router1 password cisco**

C. **interface serial 0/1/0**

D. **encapsulation ppp**

E. **encapsulation hdlc**

F. **ppp authentication pap**

G. **ppp authentication chap**

13. What is required to successfully establish a connection between two routers using CHAP authentication?

A. The hostnames of both routers must be the same.

B. The usernames of both routers must be the same.

C. The enable secret passwords configured on both routers must be the same.

D. The password configured with the router's username must be the same on both routers.

E. The **ppp chap sent-username** command must be configured the same on both routers.

14. For each characteristic, indicate whether it is associated with PAP or CHAP:

Two-way handshake

Three-way handshake

Open to trial-and-error attacks

Password sent in clear text

Periodic verification

Uses a one-way hash function

15. For each description, indicate whether it is associated with LCP or NCP:

    Negotiates link establishment parameters

    Negotiates Layer 3 protocol parameters

    Maintains/debugs a link

    Can negotiate multiple Layer 3 protocols

    Terminates a link

16. Describe four of the six types of WAN encapsulation protocols.

17. Describe the functions of LCP and NCP.

18. Describe the five configurable LCP encapsulation options.

19. Refer to the following configurations for Router R1 and Router R3:

```
hostname R1
username R1 password cisco123
!
int serial 0/0
ip address 128.0.1.1 255.255.255.0
encapsulation ppp
ppp authentication pap
```

```
hostname R3
username R1 password cisco
!
int serial 0/0
ip address 128.0.1.2 255.255.255.0
encapsulation ppp
ppp authentication CHAP
```

    Router R1 is unable to connect with Router R3. On the basis of the information presented, which configuration changes on Router R1 would correct the problem?

Challenge Questions and Activities

1. The following outputs are a result of the **debug ppp negotiation** command. Which of these outputs are the result of using PAP authentication, and which are the result of using CHAP authentication?

```
Serial1 PPP: Phase is AUTHENTICATING, by both
Serial1 PPP: Phase is AUTHENTICATING, by the peer
Serial1 PPP: Phase is AUTHENTICATING, by this end
```