Managing Serial Networks Using UUCP and PPP in Oracle® Solaris 11.1

Part No: E29000–01
October 2012

ORACLE

121010@25097

Contents

Figures

Tables

Examples

Preface

Managing Serial Networks Using UUCP and PPP in Oracle Solaris 11.1 is part of a multivolume set that covers a significant part of the Oracle Solaris system administration information. This book assumes that you have already installed the Oracle Solaris operating system, and you have set up any networking software that you plan to use.

Note – This Oracle Solaris release supports systems that use the SPARC and x86 families of processor architectures. The supported systems appear in the *Oracle Solaris OS: Hardware Compatibility Lists*. This document cites any implementation differences between the platform types.

Who Should Use This Book

This book is intended for anyone responsible for administering one or more systems that run the Oracle Solaris release. To use this book, you should have one to two years of UNIX system administration experience. Attending UNIX system administration training courses might be helpful.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Description	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

◆ ◆ ◆ CHAPTER 1

Solaris PPP 4.0 (Overview)

This section covers serial networking topics. Serial networking refers to the use of a serial interface, such as an RS-232 or V.35 port, to connect two or more computers for data transfer. Unlike LAN interfaces, such as Ethernet, these serial interfaces are used to connect systems that are separated by large distances. PPP (Point-to-Point Protocol) and UUCP (UNIX-to-UNIX CoPy) are distinct technologies that can be used to implement serial networking. When a serial interface is configured for networking, it is made available for multiple users, in much the same way as any other network interface, such as Ethernet.

This chapter introduces Solaris PPP 4.0. This version of PPP enables two computers in different physical locations to communicate with each other by using PPP over a variety of media. Solaris PPP 4.0 is included as part of the base installation.

The following topics are discussed:

Solaris PPP 4.0 Basics

Solaris PPP 4.0 implements the Point-to-Point Protocol (PPP), a data link protocol, which is a member of the TCP/IP protocol suite. PPP describes how data is transmitted between two endpoint machines, over communications media such as telephone lines.

Since the early 1990s, PPP has been a widely used Internet standard for sending datagrams over a communications link. The PPP standard is described in RFC 1661 by the Point-to-Point Working Group of the Internet Engineering Task Force (IETF). PPP is commonly used when remote computers call an Internet service provider (ISP) or a corporate server that is configured to receive incoming calls.

Solaris PPP 4.0 is based on the publicly available Australian National University (ANU) PPP–2.4 and implements the PPP standard. Both asynchronous and synchronous PPP links are supported.

Solaris PPP 4.0 Compatibility

Various versions of standard PPP are available and in wide use throughout the Internet community. ANU PPP-2.4 is a popular choice for Linux, Tru64 UNIX,and all three major BSD variants:

- FreeBSD
- OpenBSD
- NetBSD

Solaris PPP 4.0 brings the highly configurable features of ANU PPP-2.4 to machines that run the Oracle Solaris operating system. Machines that run Solaris PPP 4.0 can easily set up PPP links to any machine that runs an implementation of standard PPP.

Some non-ANU-based PPP implementations that successfully interoperate with Solaris PPP 4.0 include the following:

- Solaris PPP, also known as `asppp`, available with the Solaris 2.4 through Solaris 8 releases
- Solstice PPP 3.0.1
- Microsoft Windows 98 DUN
- Cisco IOS 12.0 (synchronous)

Which Version of Solaris PPP to Use

Solaris PPP 4.0 is the PPP implementation that is supported. The Solaris 9 release and later releases do not include the earlier Asynchronous Solaris PPP (`asppp`) software. For more information, refer to Chapter 9, "Migrating From Asynchronous Solaris PPP to Solaris PPP 4.0 (Tasks)."

Why Use Solaris PPP 4.0?

If you currently use `asppp`, consider migrating to Solaris PPP 4.0. Note the following differences between the two Solaris PPP technologies:

- **Transfer modes**

  `asppp` supports asynchronous communications only. Solaris PPP 4.0 supports both asynchronous communications and synchronous communications.

- **Configuration process**

Setting up `asppp` requires configuring the `asppp.cf` configuration file, three UUCP files, and the `ipadm` command. Moreover, you have to preconfigure interfaces for all users who might log in to a machine.

Setting up Solaris PPP 4.0 requires defining options for the PPP configuration files, or issuing the `pppd` command with options. You can also use a combination of both the configuration file and command-line methods. Solaris PPP dynamically creates and removes interfaces. You do not have to directly configure PPP interfaces for each user.

- **Solaris PPP 4.0 features not available from asppp**
  - MS-CHAPv1 and MS-CHAPv2 authentication
  - PPP over Ethernet (PPPoE), to support ADSL bridges
  - PAM authentication
  - Plug-in modules
  - IPv6 addressing
  - Data compression that uses Deflate or BSD compress
  - Microsoft client-side callback support

Solaris PPP 4.0 Upgrade Path

If you are converting an existing `asppp` configuration to Solaris PPP 4.0, you can use the translation script that is provided with this release. For complete instructions, refer to "How to Convert From `asppp` to Solaris PPP 4.0" on page 152.

Where to Go for More Information About PPP

Many resources with information about PPP can be found in print and online. The following subsections give some suggestions.

Professional Reference Books About PPP

For more information about widely used PPP implementations, including ANU PPP, refer to the following books:

- Carlson, James. *PPP Design, Implementation, and Debugging*. 2nd ed. Addison-Wesley, 2000.
- Sun, Andrew. *Using and Managing PPP*. O'Reilly & Associates, 1999.

Web Sites About PPP

Go to the following web sites for general information about PPP:

- For technical information, FAQs, discussions about Oracle Solaris system administration, and earlier versions of PPP, go to the system administrators' resource, http://www.sun.com/bigadmin/home/index.html.
- For modem configuration and advice about many different implementations of PPP, refer to Stokely Consulting's Web Project Management & Software Development web site: http://www.stokely.com/unix.serial.port.resources/ppp.slip.html.

Requests for Comments (RFCs) About PPP

Some useful Internet RFCs about PPP include the following:

- 1661 and 1662, which describe the major features of PPP
- 1334, which describes authentication protocols, such as Password Authentication Protocol (PAP) and Challenge-Handshake Authentication Protocol (CHAP)
- 1332, an informational RFC that describes PPP over Ethernet (PPPoE)

To obtain copies of PPP RFCs, specify the number of the RFC on the IETF RFC web page at http://www.ietf.org/rfc.html.

Man Pages About PPP

For technical details about the Solaris PPP 4.0 implementation, refer to the following man pages:

- `pppd`(1M)
- `chat`(1M)
- `pppstats`(1M)
- `pppoec`(1M)
- `pppoed`(1M)
- `sppptun`(1M)
- `snoop`(1M)

Also, see the man page for `pppdump`(1M). You can find the PPP-related man pages by using the `man` command.

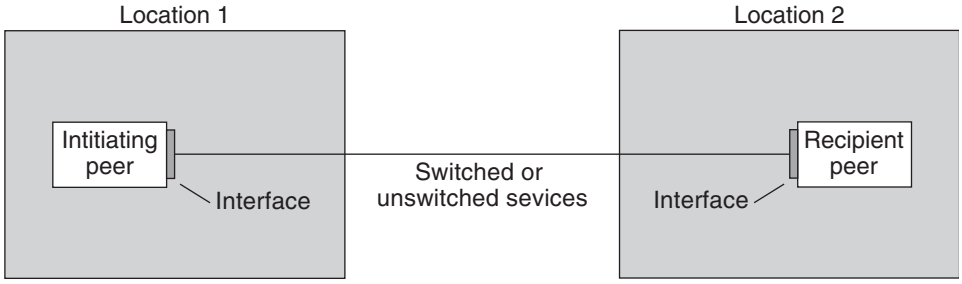
PPP Configurations and Terminology

This section introduces PPP configurations. The section also defines terms that are used in this guide.

Solaris PPP 4.0 supports a number of configurations.

- Switched-access, or *dial-up*, configurations
- Hardwired, or *leased-line* configurations

FIGURE 1–1 Parts of the PPP Link



The previous figure shows a basic PPP link. The link has the following parts:

- Two machines, usually in separate physical locations, called *peers*. A peer could be a personal computer, engineering workstation, large server, or even a commercial router, depending on a site's requirements.
- Serial interface on each peer. On Oracle Solaris machines, this interface could be `cua`, `hihp`, or other interface, depending on whether you configure asynchronous or synchronous PPP.
- Physical link, such as a serial cable, a modem connection, or a leased line from a network provider, such as a T1 or T3 line.

Dial-up PPP Overview

The most commonly used PPP configuration is the *dial-up link*. In a dial-up link, the local peer *dials up* the remote peer to establish the connection and run PPP. In the dial-up process, the local peer calls the remote peer's telephone number to initiate the link.

A common dial-up scenario includes a home computer that calls a peer at an ISP, configured to receive incoming calls. Another scenario is a corporate site where a local machine transmits data over a PPP link to a peer in another building.

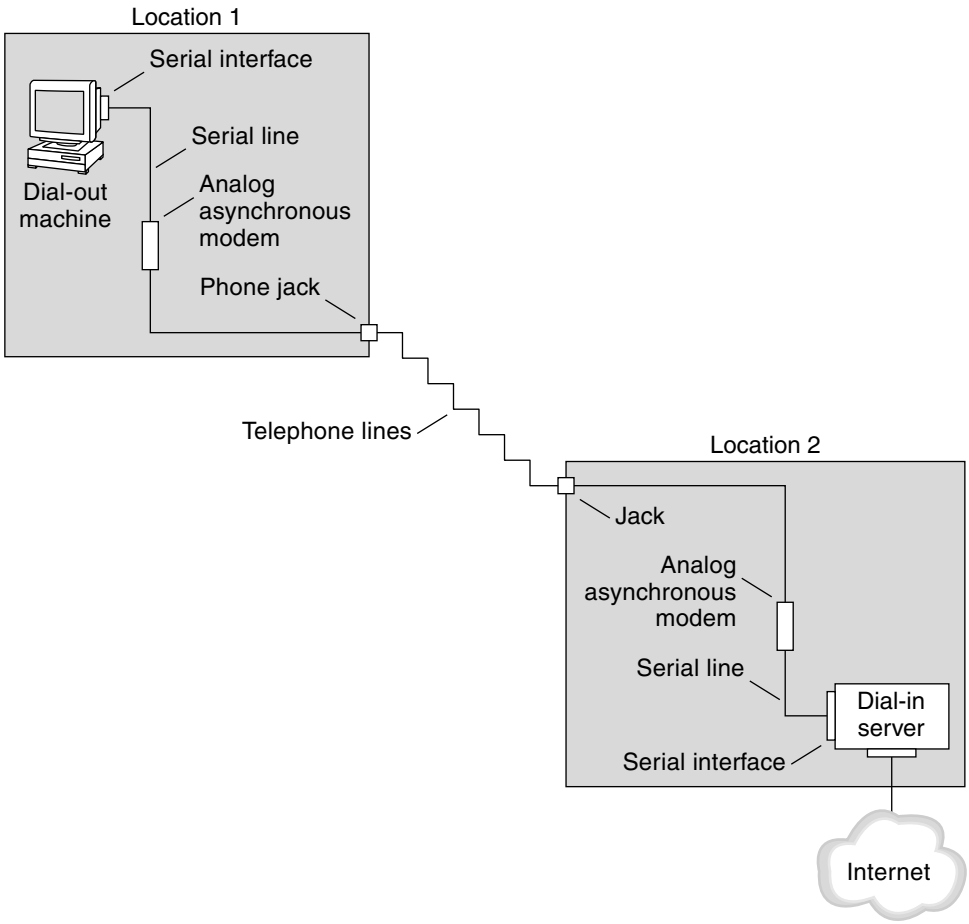
In this guide, the local peer that initiates the dial-up connection is referred to as the *dial-out machine*. The peer that receives the incoming call is referred to as the *dial-in server*. This machine is actually the target peer of the dial-out machine and might or might not be a true server.

PPP is not a client-server protocol. Some PPP documents use the terms “client” and “server” to refer to telephone call establishment. A dial-in server is not a true server like a file server or name server. Dial-in server is a widely used PPP term because dial-in machines often “serve” network accessibility to more than one dial-out machine. Nevertheless, the dial-in server is the target peer of the dial-out machine.

Parts of the Dial-up PPP Link

See the following figure.

FIGURE 1–2 Basic Analog Dial-up PPP Link



The configuration for Location 1, the dial-out side of the link, is composed of the following elements:

- Dial-out machine, typically a personal computer or workstation in an individual's home.
- Serial interface on the dial-out machine. `/dev/cua/a` or `/dev/cua/b` is the standard serial interface for outgoing calls on machines that run Oracle Solaris software.
- Asynchronous modem or ISDN terminal adapter (TA) that is connected to a telephone jack.
- Telephone lines and services of a telephone company.

The configuration for Location 2, the dial-in side of the link, is composed of the following elements:

- Telephone jack or similar connector, which is connected to the telephone network
- Asynchronous modem or ISDN TA
- Serial interface on the dial-in server, either `ttya` or `ttyb` for incoming calls
- Dial-in server, which is connected to a network, such as a corporate intranet, or, in the instance of an ISP, the global Internet

Using ISDN Terminal Adapters With a Dial-out Machine

External ISDN TAs have faster speeds than modems, but you configure TAs in basically the same way. The major difference in configuring an ISDN TA is in the chat script, which requires commands specific to the TA's manufacturer. Refer to "Chat Script for External ISDN TA" on page 128 for information about chat scripts for ISDN TAs.

What Happens During Dial-up Communications

PPP configuration files on both the dial-out and dial-in peers contain instructions for setting up the link. The following process occurs as the dial-up link is initiated.

1. User or process on the dial-out machine runs the `pppd` command to start the link.
2. Dial-out machine reads its PPP configuration files. The dial-out machine then sends instructions over the serial line to its modem, including the phone number of the dial-in server.
3. Modem dials the phone number to establish a telephone connection with the modem on the dial-in server.

   The series of text strings that the dial-out machine sends to the modem and dial-in server are contained in a file called a *chat script*. If necessary, the dial-out machine sends commands to the dial-in server to invoke PPP on the server.
4. Modem attached to the dial-in server begins link negotiation with the modem on the dial-out machine.
5. When modem-to-modem negotiation is completed, the modem on the dial-out machine reports "CONNECT."
6. PPP on both peers enters *Establish* phase, where Link Control Protocol (LCP) negotiates basic link parameters and the use of authentication.
7. If necessary, the peers authenticate each other.
8. PPP's Network Control Protocols (NCPs) negotiate the use of network protocols, such as IPv4 or IPv6.

The dial-out machine can then run `telnet` or a similar command to a host that is reachable through the dial-in server.

Leased-Line PPP Overview

A hardwired, *leased-line* PPP configuration involves two peers that are connected by a link. This link consists of a switched or an unswitched digital service leased from a provider. Solaris PPP 4.0 works over any full-duplex, point-to-point leased-line medium. Typically, a company rents a hardwired link from a network provider to connect to an ISP or other remote site.

Comparison of Dial-up and Leased-Line Links

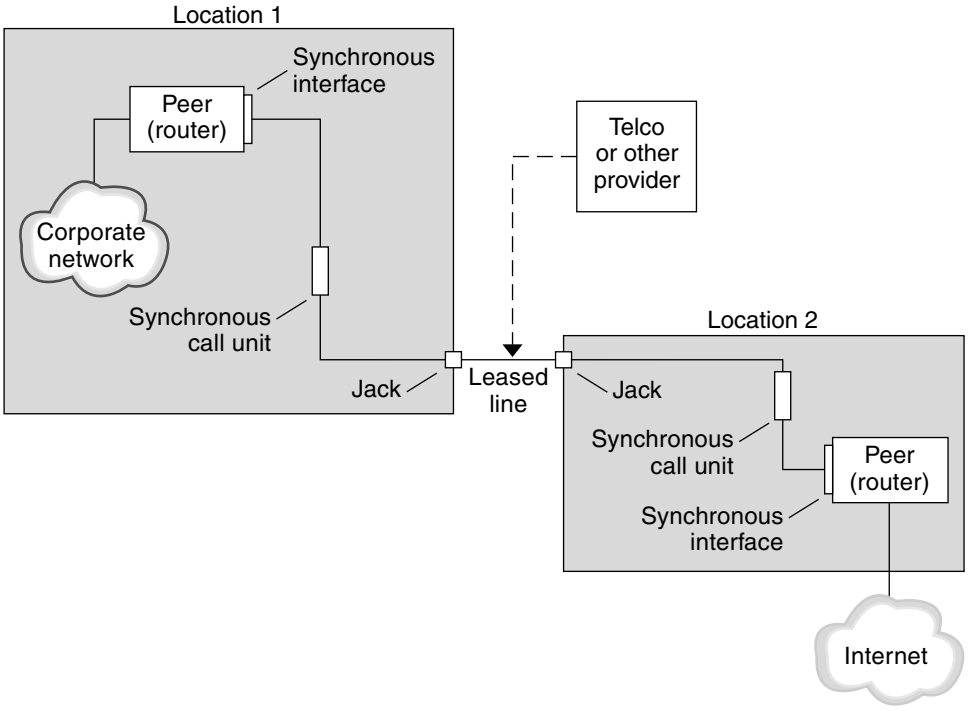
Both dial-up and leased-line links involve two peers that are connected by a communications medium. The next table summarizes the differences between the link types.

Leased Line	Dial-up Line
Always connected, unless a system administrator or power failure takes the leased-line down.	Initiated on demand, when a user tries to call a remote peer.
Uses synchronous and asynchronous communications. For asynchronous communications, a long-haul modem is often used.	Uses asynchronous communications.
Rented from a provider.	Uses existing telephone lines.
Requires synchronous units.	Uses less costly modems.
Requires synchronous ports, which are common on most SPARC systems. However, synchronous ports are not common on x86 systems and newer SPARC systems.	Uses standard serial interfaces that are included on most computers.

Parts of a Leased-Line PPP Link

See the following figure.

FIGURE 1–3 Basic Leased-Line Configuration



The leased-line link contains the following parts:

- **Two peers**, each peer at one end of the link. Each peer might be a workstation or server. Often the peer functions as a router between its network or the Internet, and the opposite peer.
- **Synchronous interface on each peer**. Some machines that run Oracle Solaris software require you to purchase a synchronous interface card, such as HSI/P, to connect to a leased line. Other machines, such as UltraSPARC workstations, have built-in synchronous interfaces.
- **CSU/DSU synchronous digital unit on each peer**, which connects the synchronous port to the leased line.

  A CSU might be built-in to the DSU, or owned by you, or leased from a provider, depending on your locale. The DSU gives the Oracle Solaris machine a standard synchronous serial interface. With Frame Relay, the Frame Relay Access Device (FRAD) performs the serial interface adaptation.
- **Leased line**, providing switched or unswitched digital services. Some examples are SONET/SDH, Frame Relay PVC, and T1.

What Happens During Leased-Line Communications

On most types of leased lines, peers do not actually dial each other. Rather, a company purchases a leased-line service to connect explicitly between two fixed locations. Sometimes the two peers at either end of the leased line are at different physical locations of the same company. Another scenario is a company that sets up a router on a leased line that is connected to an ISP.

Leased lines are less commonly used than dial-up links, though the hardwired links are easier to set up. Hardwired links do not require chat scripts. Authentication is often not used because both peers are known to each other when a line is leased. After the two peers initiate PPP over the link, the link stays active. A leased-line link remains active unless the line fails, or either peer explicitly terminates the link.

A peer on a leased line that runs Solaris PPP 4.0 uses most of the same configuration files that define a dial-up link.

The following process occurs to initiate communication over the leased line:

1. Each peer machine runs the `pppd` command as part of the booting process or another administrative script.
2. The peers read their PPP configuration files.
3. The peers negotiate communications parameters.
4. An IP link is established.

PPP Authentication

Authentication is the process of verifying that a user is who he or she claims to be. The UNIX login sequence is a simple form of authentication:

1. The `login` command prompts the user for a name and password.
2. `login` then attempts to authenticate the user by looking up the typed user name and password in the password database.
3. If the database contains the user name and password, then the user is *authenticated* and given access to the system. If the database does not contain the user name and password, the user is denied access to the system.

By default, Solaris PPP 4.0 does not demand authentication on machines that do not have a default route specified. Thus, a local machine without a default route does not authenticate remote callers. Conversely, if a machine does have a default route defined, the machine always authenticates remote callers.

You might use PPP authentication protocols to verify the identity of callers who are trying to set up a PPP link to your machine. Conversely, you must configure PPP authentication information if your local machine must call peers that authenticate callers.

Authenticators and Authenticatees

The calling machine on a PPP link is considered the *authenticatee* because the caller must prove its identity to the remote peer. The peer is considered the *authenticator*. The authenticator looks up the caller's identity in the appropriate PPP files for the security protocol and authenticates or does not authenticate the caller.

You typically configure PPP authentication for a dial-up link. When the call begins, the dial-out machine is the authenticatee. The dial-in server is the authenticator. The server has a database in the form of a *secrets* file. This file lists all users who are granted permission to set up a PPP link to the server. Think of these users as *trusted callers*.

Some dial-out machines require remote peers to provide authentication information when responding to the dial-out machine's call. Then their roles are reversed: the remote peer becomes the authenticatee and the dial-out machine the authenticator.

Note – PPP 4.0 does not prevent authentication by leased-line peers, but authentication is not often used in leased-line links. The nature of leased-line contracts usually means that both participants on the ends of the line are known to each other. Both participants often are trusted. However, because PPP authentication is not that difficult to administer, you should seriously consider implementing authentication for leased lines.

PPP Authentication Protocols

The PPP authentication protocols are Password Authentication Protocol (PAP) and Challenge-Handshake Authentication Protocol (CHAP). Each protocol uses a *secrets* database that contains identification information, or *security credentials*, for each caller that is permitted to link to the local machine. For a detailed explanation of PAP, see "Password Authentication Protocol (PAP)" on page 131. For a CHAP explanation, see "Challenge-Handshake Authentication Protocol (CHAP)" on page 134.

Why Use PPP Authentication?

Providing authentication on a PPP link is optional. Moreover, though authentication does verify that a peer is to be trusted, PPP authentication does not provide confidentiality of data. For confidentiality, use encryption software, such as IPsec, PGP, SSL, Kerberos, and the Secure Shell.

Note – Solaris PPP 4.0 does not implement the PPP Encryption Control Protocol (ECP), which is described in RFC 1968.

Consider implementing PPP authentication in the following situations:

- Your company accepts incoming calls from users over the public, switched telephone network.
- Your corporate security policy requires remote users to provide authentication credentials when accessing your network through a corporate firewall or when engaging in secure transactions.
- You want to authenticate callers against a standard UNIX password database, such as `/etc/passwd`, NIS, LDAP, or PAM. Use PAP authentication for this scenario.
- Your company's dial-in servers also provide the network's Internet connection. Use PAP authentication for this scenario.
- The serial line is less secure than the password database on the machine or networks at either end of the link. Use CHAP authentication for this scenario.

Support for DSL Users Through PPPoE

Many network providers and individuals who are working at home use Digital Subscriber Line (DSL) technology to provide fast network access. To support DSL users, Solaris PPP 4.0 includes the PPP over Ethernet (PPPoE) feature. PPPoE technology enables multiple hosts to run PPP sessions over one Ethernet link to one or more destinations.

If one of the following factors applies to your situation, you should use PPPoE:

- You support DSL users, possibly including yourself. Your DSL service provider might require users to configure a PPPoE tunnel to receive services over the DSL line.
- Your site is an ISP that intends to offer PPPoE to customers.

This section introduces terms that are associated with PPPoE and an overview of a basic PPPoE topology.

PPPoE Overview

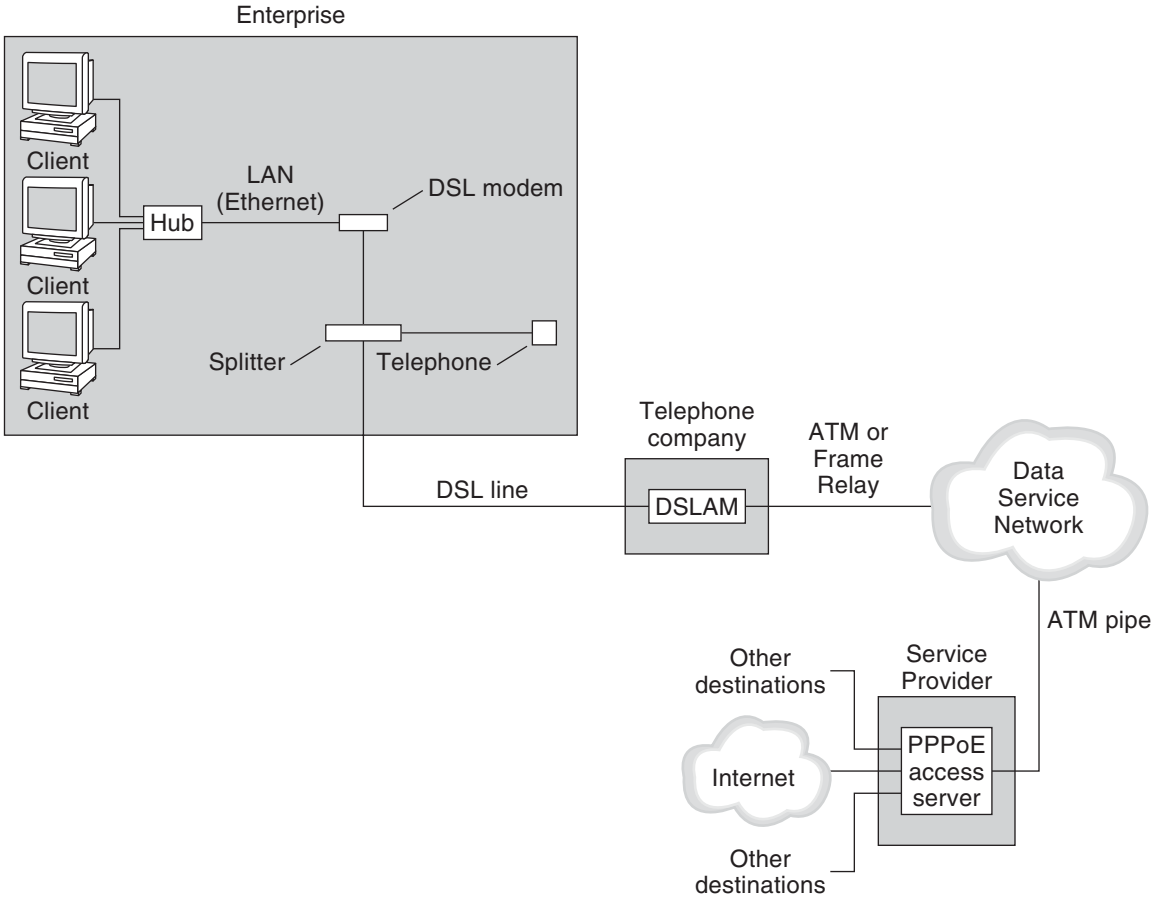
PPPoE is a proprietary protocol from RedBack Networks. PPPoE is a discovery protocol, rather than another version of standard PPP. In a PPPoE scenario, a machine that initiates PPP communications first must locate, or *discover*, a peer that runs PPPoE. The PPPoE protocol uses Ethernet broadcast packets to locate the peer.

After the discovery process, PPPoE sets up an Ethernet-based tunnel from the initiating host, or *PPPoE client*, to the peer, the *PPPoE access server*. *Tunneling* is the practice of running one protocol on top of another protocol. Using PPPoE, Solaris PPP 4.0 tunnels PPP over Ethernet IEEE 802.2, both of which are data link protocols. The resulting PPP connection behaves like a dedicated link between the PPPoE client and the access server. For detailed information about PPPoE, see "Creating PPPoE Tunnels for DSL Support" on page 139.

Parts of a PPPoE Configuration

Three participants are involved in a PPPoE configuration: a consumer, a telephone company, and a service provider, as the following figure shows.

FIGURE 1–4 Participants in a PPPoE Tunnel



PPPoE Consumers

As system administrator, you might assist consumers with their PPPoE configurations. One common type of PPPoE consumer is an individual who needs to run PPPoE over a DSL line. Another PPPoE consumer is a company that purchases a DSL line through which employees can run PPPoE tunnels, as illustrated in the previous figure.

The main reason for a corporate consumer to use PPPoE is to offer PPP communications through a high-speed DSL device to a number of hosts. Often, a single PPPoE client has an individual *DSL modem*. Or, a group of clients on a hub might share a DSL modem that is also connected to the hub by an Ethernet line.

Note – DSL devices are technically bridges, not modems. However, because common practice is to refer to these devices as modems, this guide uses the term "DSL modem."

PPPoE runs PPP over a tunnel on the Ethernet line that is connected to the DSL modem. That line is connected to a splitter, which, in turn connects to a telephone line.

PPPoE at a Telephone Company

The telephone company is the middle layer of the PPPoE scenario. The telephone company splits the signal that is received over the phone line by using a device that is called a *Digital Subscriber Line Access Multiplexer (DSLAM)*. The DSLAM breaks out the signals onto separate wires, analog wires for telephone service, and digital wires for PPPoE. From the DSLAM, the digital wires extend the tunnel over an ATM data network to the ISP.

PPPoE at a Service Provider

The ISP receives the PPPoE transmission from the ATM data network over a bridge. At the ISP, an access server that runs PPPoE functions as the peer for the PPP link. The access server is very similar in function to the dial-in server that was introduced in Figure 1–2, but the access server does not use modems. The access server converts the individual PPPoE sessions into regular IP traffic, for example Internet access.

If you are a system administrator for an ISP, you might be responsible for configuring and maintaining an access server.

Security on a PPPoE Tunnel

The PPPoE tunnel is inherently insecure. You can use PAP or CHAP to provide user authentication for the PPP link that is running over the tunnel.

◆ ◆ ◆ CHAPTER 2

Planning for the PPP Link (Tasks)

Setting up a PPP link involves a set of discrete tasks, which includes planning tasks and other activities that are not related to PPP. This chapter explains how to plan for the most common PPP links, for authentication, and for PPPoE.

The task chapters that follow Chapter 2, "Planning for the PPP Link (Tasks)," use sample configurations to illustrate how to set up a particular link. These sample configurations are introduced in this chapter.

Topics that are covered include the following:

- "Planning a Dial-up PPP Link" on page 36
- "Planning a Leased-Line Link" on page 39
- "Planning for Authentication on a Link" on page 41
- "Planning for DSL Support Over a PPPoE Tunnel" on page 47

Overall PPP Planning (Task Map)

PPP requires planning tasks before you actually can set up the link. Moreover, if you want to use a PPPoE tunneling, you first have to set up the PPP link and then provide tunneling. The following task map lists the large planning tasks that are discussed in this chapter. You might need to use only the general task for the link type to be configured. Or you might require the task for the link, authentication, and perhaps PPPoE.

TABLE 2–1 Task Map for PPP Planning

Task	Description	For Instructions
Plan for a dial-up PPP link	Gather information that is required to set up a dial-out machine or a dial-in server	"Planning a Dial-up PPP Link" on page 36
Plan for a leased-line link	Gather information that is required to set up a client on a leased line	"Planning a Leased-Line Link" on page 39

TABLE 2–1 Task Map for PPP Planning *(Continued)*

Task	Description	For Instructions
Plan for authentication on the PPP link	Gather information that is required to configure PAP or CHAP authentication on the PPP link	"Planning for Authentication on a Link" on page 41
Plan for a PPPoE tunnel	Gather information that is required to set up a PPPoE tunnel over which a PPP link can run	"Planning for DSL Support Over a PPPoE Tunnel" on page 47

Planning a Dial-up PPP Link

Dial-up links are the most commonly used PPP links. This section includes the following information:

- Planning information for a dial-up link
- Explanation of the sample link to be used in Chapter 3, "Setting Up a Dial-up PPP Link (Tasks)"

Typically, you only configure the machine at one end of the dial-up PPP link, the dial-out machine, or the dial-in server. For an introduction to dial-up PPP, refer to "Dial-up PPP Overview" on page 23.

Before You Set Up the Dial-out Machine

Before you configure a dial-out machine, gather the information that is listed in the following table.

Note – The planning information in this section does not include information to be gathered about authentication or PPPoE. For details about authentication planning, refer to "Planning for Authentication on a Link" on page 41. For PPPoE planning, refer to "Planning for DSL Support Over a PPPoE Tunnel" on page 47.

TABLE 2–2 Information for a Dial-out Machine

Information	Action
Maximum modem speed	Refer to documentation that was provided by the modem manufacturer.
Modem connection commands (AT commands)	Refer to documentation that was provided by the modem manufacturer.
Name to use for dial-in server at the other end of the link	Create any name that helps you identify the dial-in server.
Login sequence that was required by dial-in server	Contact the dial-in server's administrator or ISP documentation if dial-in server is at the ISP.

Before You Set Up the Dial-in Server

Before you configure a dial-in server, gather the information that is listed in the following table.

Note – The planning information in this section does not include information to be gathered about authentication or PPPoE. For details about authentication planning, refer to "Planning for Authentication on a Link" on page 41. For PPPoE planning, refer to "Planning for DSL Support Over a PPPoE Tunnel" on page 47.

TABLE 2–3 Information for a Dial-in Server

Information	Action
Maximum modem speed	Refer to documentation that was provided by the modem manufacturer.
User names of people who are permitted to call the dial-in server	Obtain the names of the prospective users before you set up their home directories, as discussed in "How to Configure Users of the Dial-in Server" on page 60.
Dedicated IP address for PPP communications	Obtain an address from the individual at your company who is responsible for delegating IP addresses.

Example of a Configuration for Dial-up PPP

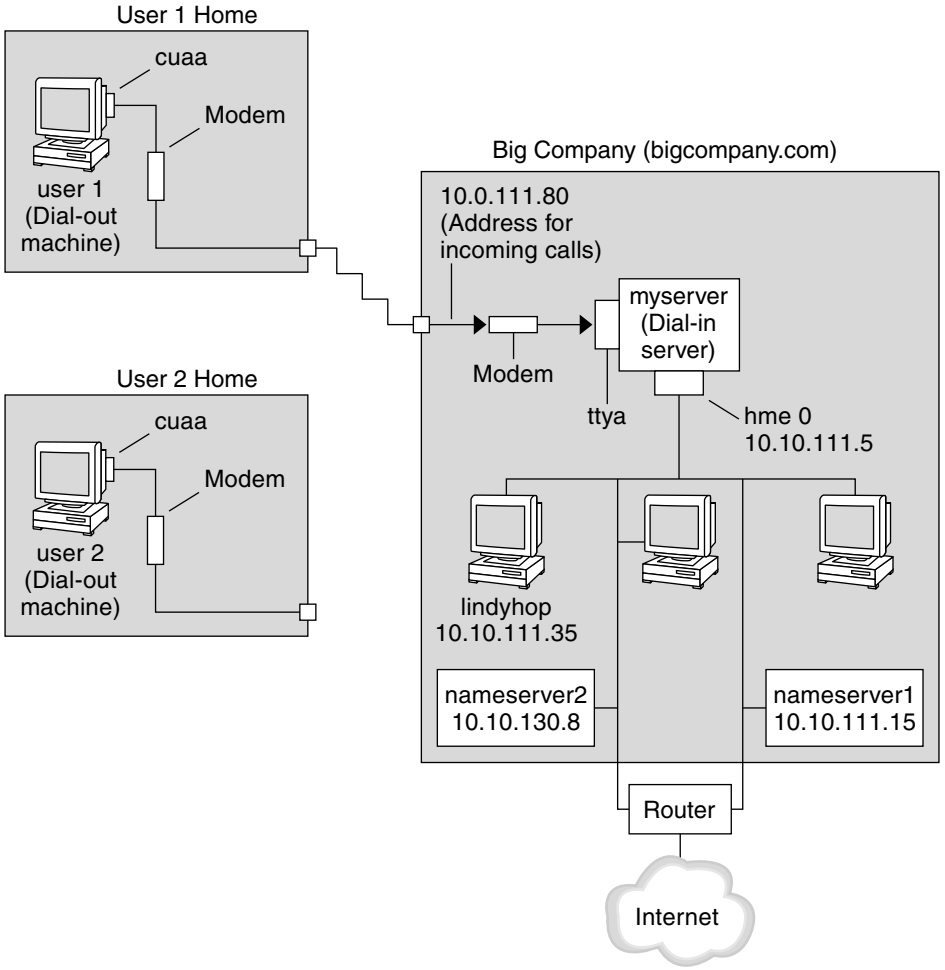
The tasks to be introduced in Chapter 3, "Setting Up a Dial-up PPP Link (Tasks)," execute a small company's requirement to let employees work at home a few days a week. Some employees require the Oracle Solaris OS on their home machines. These workers also need to log in remotely to their work machines on the corporate Intranet.

The tasks set up a basic dial-up link with the following features:

- The *dial-out* machines are at the houses of employees who need to call the corporate intranet.
- The *dial-in* server is a machine on the corporate intranet that is configured to receive incoming calls from employees.
- UNIX-style login is used to authenticate the dial-out machine. Stronger Solaris PPP 4.0 authentication methods are not required by the company's security policy.

The next figure shows the link that is set up in Chapter 3, "Setting Up a Dial-up PPP Link (Tasks)."

FIGURE 2–1 Sample Dial-up Link



In this figure, a remote host dials out through its modem over telephone lines to Big Company's Intranet. Another host is configured to dial out to Big Company but currently is inactive. The calls from remote users are answered in the order received by the modem that is attached to the dial-in server at Big Company. A PPP connection is established between the peers. The dial-out machine can then remotely log in to a host machine on the Intranet.

Where to Go for More Information About Dial-up PPP

Refer to the following:

- To set up a dial-out machine, see Table 3–2.
- To set up a dial-in machine, see Table 3–3.
- To get an overview of dial-up links, see "Dial-up PPP Overview" on page 23.
- To get detailed information about PPP files and commands, see "Using PPP Options in Files and on the Command Line" on page 111.

Planning a Leased-Line Link

Setting up a leased-line link involves configuring the peer at one end of a switched or unswitched service that is leased from a provider.

This section includes the following information:

- Planning information for a leased-line link
- Explanation of the sample link that is shown in Figure 2–2

For an introduction to leased-line links, refer to "Leased-Line PPP Overview" on page 27. For tasks about setting up the leased line, see Chapter 4, "Setting Up a Leased-Line PPP Link (Tasks)."

Before You Set Up the Leased-Line Link

When your company rents a leased-line link from a network provider, you typically configure only the system at your end of the link. The peer at the other end of the link is maintained by another administrator. This individual might be a system administrator at a remote location in your company or a system administrator at an ISP.

Hardware That Is Needed for a Leased-Line Link

In addition to the link media, your end of the link requires the following hardware:

- Synchronous interface for your system
- Synchronous unit (CSU/DSU)
- Your system

Some network providers include a router, synchronous interface, and a CSU/DSU as part of the customer premises equipment (CPE). However, necessary equipment varies, based on the provider and any governmental restrictions in your locale. The network provider can give you information about the unit that is needed, if this equipment is not provided with the leased line.

Information to Be Gathered for the Leased-Line Link

Before you configure the local peer, you might need to gather the items that are listed in the next table.

TABLE 2–4 Planning for a Leased-Line Link

Information	Action
Device name of the interface	Refer to the interface card documentation.
Configuration instructions for the synchronous interface card	Refer to the interface card documentation. You need this information to configure the HSI/P interface. You might not need to configure other types of interface cards.
(Optional) IP address of the remote peer	Refer to the service provider documentation. Alternatively, contact the system administrator of the remote peer. This information is needed only if the IP address is not negotiated between the two peers.
(Optional) Name of the remote peer	Refer to the service provider documentation. Alternatively, you can contact the system administrator of the remote peer.
(Optional) Speed of the link	Refer to the service provider documentation. Alternatively, you can contact the system administrator of the remote peer.
(Optional) Compression that is used by the remote peer	Refer to the service provider documentation. Alternatively, you can contact the system administrator of the remote peer.

Example of a Configuration for a Leased-Line Link

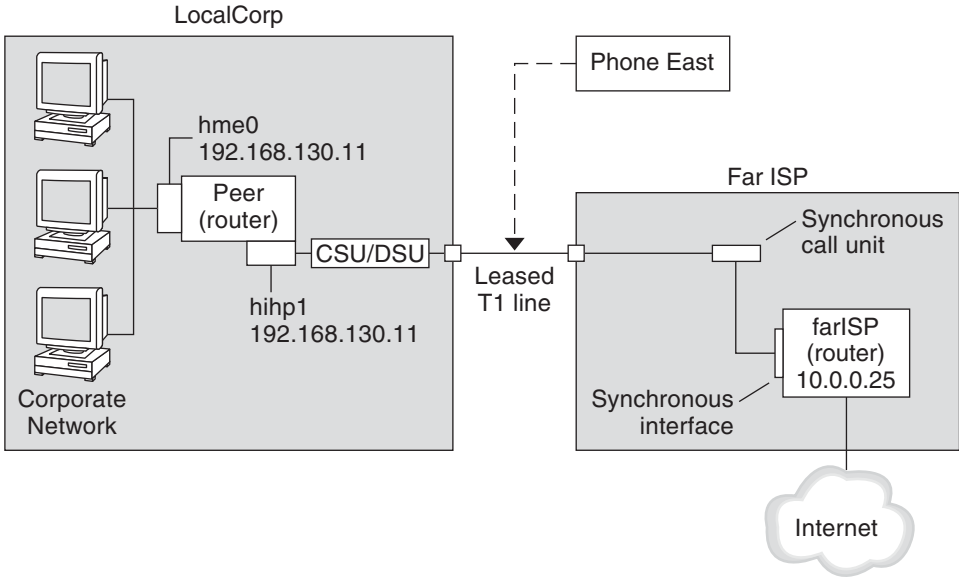
The tasks in Chapter 4, "Setting Up a Leased-Line PPP Link (Tasks)," show how to implement the goal of a medium-sized organization (LocalCorp) to provide Internet access for its employees. Currently, the employees' computers are connected on a private corporate intranet.

LocalCorp requires speedy transactions and access to the many resources on the Internet. The organization signs a contract with Far ISP, a service provider, which allows LocalCorp to set up its own leased line to Far ISP. Then, LocalCorp leases a T1 line from Phone East, a telephone company. Phone East puts in the leased line between LocalCorp and Far ISP. Then, Phone East provides a CSU/DSU that is already configured to LocalCorp.

The tasks set up a leased-line link with the following characteristics.

- LocalCorp has set up a system as a gateway router, which forwards packets over the leased line to hosts on the Internet.
- Far ISP also has set up a peer as a router to which leased lines from customers are attached.

FIGURE 2–2 Example of a Leased-Line Configuration



In the figure, a router is set up for PPP at LocalCorp. The router connects to the corporate Intranet through its `hme0` interface. The second connection is through the machine's HSI/P interface (hihp1) to the CSU/DSU digital unit. The CSU/DSU then connects to the installed leased line. The administrator at LocalCorp configures the HSI/P interface and PPP files. The administrator then types `/etc/init.d/pppd` to initiate the link between LocalCorp and Far ISP.

Where to Go for More Information About Leased Lines

Refer to the following:

- Chapter 4, "Setting Up a Leased-Line PPP Link (Tasks)"
- "Leased-Line PPP Overview" on page 27

Planning for Authentication on a Link

This section contains planning information for providing authentication on the PPP link. Chapter 5, "Setting Up PPP Authentication (Tasks)," contains tasks for implementing PPP authentication at your site.

PPP offers two types of authentication, PAP, which is described in detail in "Password Authentication Protocol (PAP)" on page 131 and CHAP, which is described in "Challenge-Handshake Authentication Protocol (CHAP)" on page 134.

Before you set up authentication on a link, you must choose which authentication protocol best meets your site's security policy. Then, you set up the secrets file and PPP configuration files for the dial-in machines, or callers' dial-out machines, or both types of machines. For information about choosing the appropriate authentication protocol for your site, see "Why Use PPP Authentication?" on page 30.

This section includes the following information:

- Planning information for both PAP and CHAP authentication
- Explanations of the sample authentication scenarios that are shown in Figure 2–3 and Figure 2–4

For tasks about setting up authentication, see Chapter 5, "Setting Up PPP Authentication (Tasks)."

Before You Set Up PPP Authentication

Setting up authentication at your site should be an integral part of your overall PPP strategy. Before implementing authentication, you should assemble the hardware, configure the software, and test the link.

TABLE 2–5 Prerequisites Before Configuring Authentication

Information	For Instructions
Tasks for configuring a dial-up link	Chapter 3, "Setting Up a Dial-up PPP Link (Tasks)."
Tasks for testing the link	Chapter 7, "Fixing Common PPP Problems (Tasks)."
Security requirements for your site	Your corporate security policy. If you do not have a policy, setting up PPP authentication gives you an opportunity to create a security policy.
Suggestions about whether to use PAP or CHAP at your site	"Why Use PPP Authentication?" on page 30. For more detailed information about these protocols, refer to "Authenticating Callers on a Link" on page 131.

Examples of PPP Authentication Configurations

This section contains examples of authentication scenarios to be used in the procedures in Chapter 5, "Setting Up PPP Authentication (Tasks)."

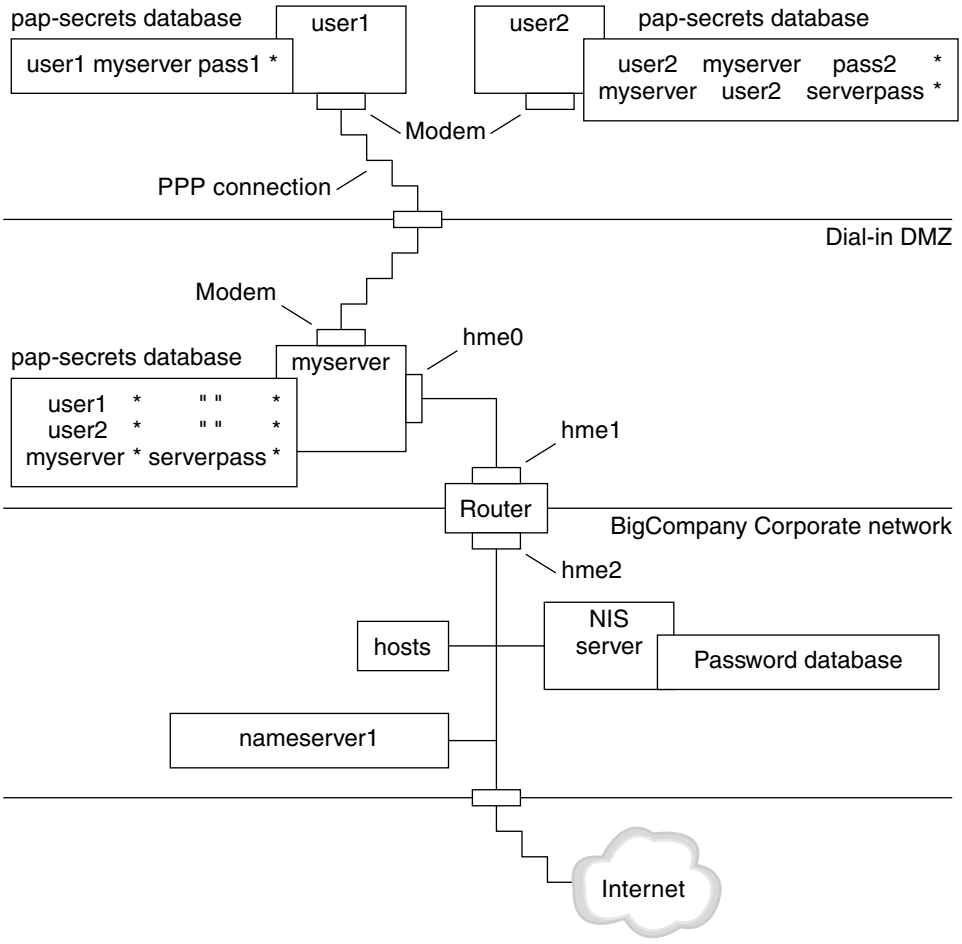
- "Example of a Configuration Using PAP Authentication" on page 43
- "Example of a Configuration Using CHAP Authentication" on page 45

Example of a Configuration Using PAP Authentication

The tasks in "Configuring PAP Authentication" on page 72 show how to set up PAP authentication over the PPP link. The procedures use as an example a PAP scenario that was created for the fictitious "Big Company" in "Example of a Configuration for Dial-up PPP" on page 37.

Big Company wants to enable its users to work from home. The system administrators want a secure solution for the serial lines to the dial-in server. UNIX-style login that uses the NIS password databases has served Big Company's network well in the past. The system administrators want a UNIX-like authentication scheme for calls that come in to the network over the PPP link. So, the administrators implement the following scenario that uses PAP authentication.

FIGURE 2–3 Example of a PAP Authentication Scenario (Working From Home)



The system administrators create a dedicated dial-in DMZ that is separated from the rest of the corporate network by a router. The term DMZ comes from the military term "demilitarized zone." The DMZ is an isolated network that is set up for security purposes. The DMZ typically contains resources that a company offers to the public, such as web servers, anonymous FTP servers, databases, and modem servers. Network designers often place the DMZ between a firewall and a company's Internet connection.

The only occupants of the DMZ that is pictured in Figure 2–3 are the dial-in server `myserver` and the router. The dial-in server requires callers to provide PAP credentials, including user names and passwords, when setting up the link. Furthermore, the dial-in server uses the `login` option of PAP. Therefore, the callers' PAP user names and passwords must correspond exactly to their UNIX user names and passwords in the dial-in server's password database.

After the PPP link is established, the caller's packets are forwarded to the router. The router forwards the transmission to its destination on the corporate network or on the Internet.

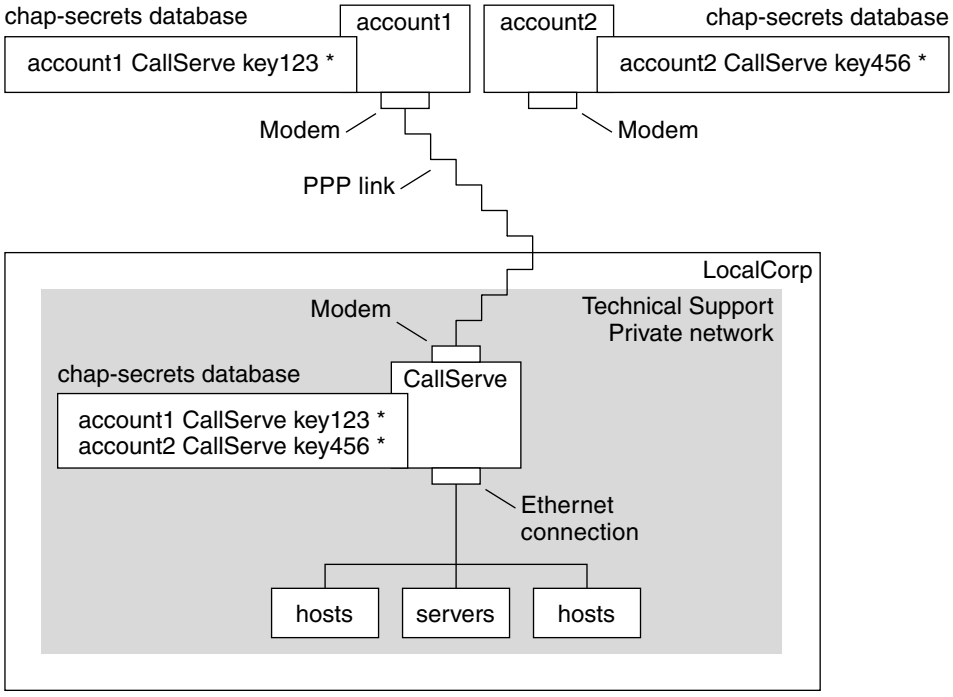
Example of a Configuration Using CHAP Authentication

The tasks in "Configuring CHAP Authentication" on page 79 show how to set up CHAP authentication. The procedures use as an example a CHAP scenario to be created for the fictitious LocalCorp that was introduced in "Example of a Configuration for a Leased-Line Link" on page 40.

LocalCorp provides connectivity to the Internet over a leased line to an ISP. The Technical Support department within LocalCorp generates heavy network traffic. Therefore, Technical Support requires its own, isolated private network. The department's field technicians travel extensively and need to access the Technical Support network from remote locations for problem-solving information. To protect sensitive information in the private network's database, remote callers must be authenticated in order to be granted permission to log in.

Therefore, the system administrators implement the following CHAP authentication scenario for a dial-up PPP configuration.

FIGURE 2–4 Example of a CHAP Authentication Scenario (Calling a Private Network)



The only link from the Technical Support network to the outside world is the serial line to the dial-in server's end of the link. The system administrators configure the laptop computer of each field service representative for PPP with CHAP security, including a CHAP secret. The chap-secrets database on the dial-in server contains the CHAP credentials for all machines that are allowed to call in to the Technical Support network.

Where to Go for More Information About Authentication

Choose from the following:

- See "Configuring PAP Authentication" on page 72.
- See "Configuring CHAP Authentication" on page 79.
- See "Authenticating Callers on a Link" on page 131 and the pppd(1M) man page.

Planning for DSL Support Over a PPPoE Tunnel

Some DSL providers require you to set up PPPoE tunneling for your site in order to run PPP over the providers' DSL lines and high-speed digital networks. For an overview of PPPoE, see "Support for DSL Users Through PPPoE" on page 31.

A PPPoE tunnel involves three participants: a consumer, a telephone company, and an ISP. You either configure PPPoE for consumers, such as PPPoE clients at your company or consumers in their homes, or you configure PPPoE on a server at an ISP.

This section contains planning information for running PPPoE on both clients and access servers. The following topics are covered:

- Planning information for the PPPoE host and access server
- Explanation of the PPPoE scenario that is introduced in "Example of a Configuration for a PPPoE Tunnel" on page 49

For tasks about setting up a PPPoE tunnel, see Chapter 6, "Setting Up a PPPoE Tunnel (Tasks)."

Before You Set Up a PPPoE Tunnel

Your preconfiguration activities depend on whether you configure the client side or server side of the tunnel. In either instance, you or your organization must contract with a telephone company. The telephone company provides the DSL lines for clients, and some form of bridging and possibly an ATM pipe for access servers. In most contracts, the telephone company assembles its equipment at your site.

Before Configuring a PPPoE Client

PPPoE client implementations usually consist of the following equipment:

- Personal computer or other system that is used by an individual
- DSL modem, which is usually installed by the telephone company or Internet access provider
- (Optional) A hub, if more than one client is involved, as is true for corporate DSL consumers
- (Optional) A splitter, usually installed by the provider

Many different DSL configurations are possible, which depend on the user or corporation's needs and the services that are offered by the provider.

TABLE 2–6 Planning for PPPoE Clients

Information	Action
If setting up a home PPPoE client for an individual or yourself, get any setup information that is outside the scope of PPPoE.	Ask the telephone company or ISP for any required setup procedures.
If setting up PPPoE clients at a corporate site, gather the names of users who are being assigned PPPoE client systems. If you configure remote PPPoE clients, you might be responsible for giving users information about adding home DSL equipment.	Ask management at your company for a list of authorized users.
Find out which interfaces are available on the PPPoE client.	Run the `ipadm show-addr` command on each machine for interface names.
(Optional) Obtain the password for the PPPoE client.	Ask users for their preferred passwords. Or, assign passwords to the users. Note that this password is used for link authentication, not for UNIX login.

Before Configuring a PPPoE Server

Planning for a PPPoE access server involves working with the telephone company that provides your connection to its data service network. The telephone company installs its lines, often ATM pipes, at your site, and provides some sort of bridging into your access server. You need to configure the Ethernet interfaces that access the services that your company provides. For example, you need to configure interfaces for Internet access, as well as the Ethernet interfaces from the telephone company's bridge.

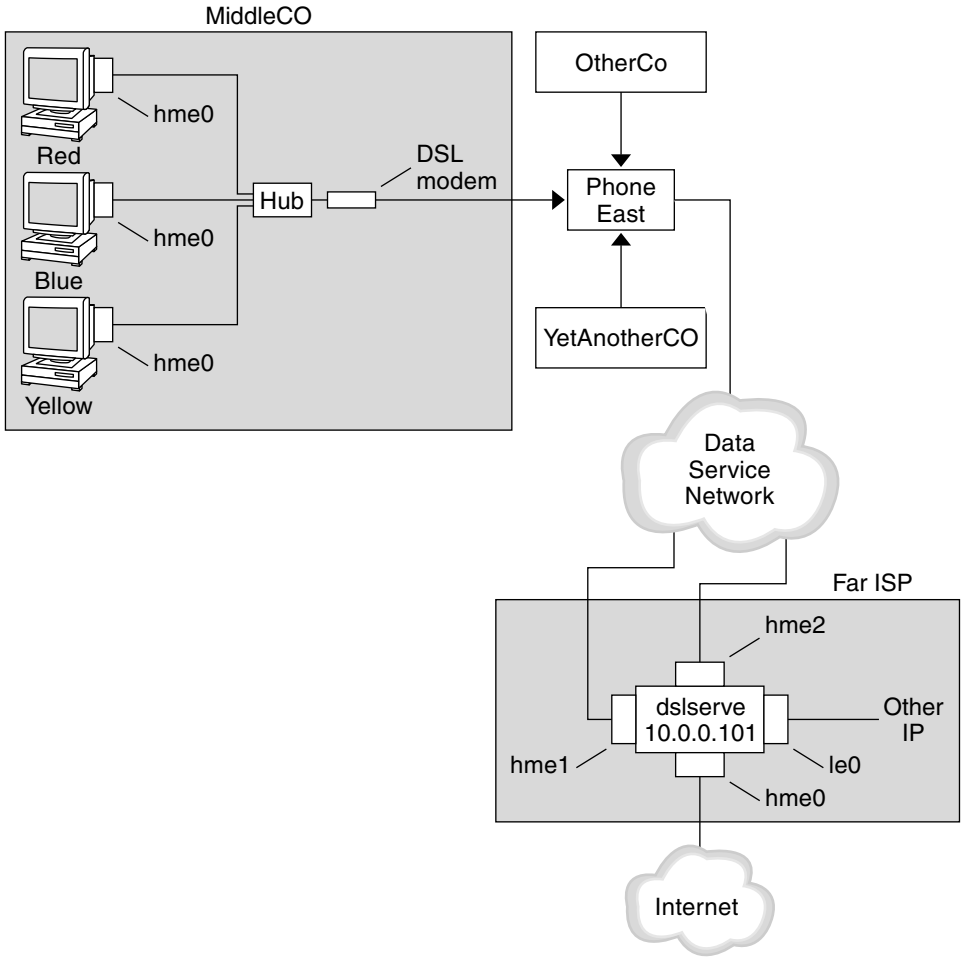
TABLE 2–7 Planning for a PPPoE Access Server

Information	Action
Interfaces that are used for lines from data service network	Run the `ipadm show-addr` command to identify interfaces.
Types of services to provide from the PPPoE server	Ask management and network planners for their requirements and suggestions.
(Optional) Types of services to provide to the consumers	Ask management and network planners for their requirements and suggestions.
(Optional) Host names and passwords for remote clients	Ask network planners and other individuals at your site who are responsible for contract negotiations. The host names and passwords are used for PAP or CHAP authentication, not for UNIX login.

Example of a Configuration for a PPPoE Tunnel

This section contains an example of a PPPoE tunnel, which is used as an illustration for the tasks in Chapter 6, "Setting Up a PPPoE Tunnel (Tasks)." Though the illustration shows all participants in the tunnel, you only administer one end, either the client side or server side.

FIGURE 2–5 Example of a PPPoE Tunnel



In the sample, MiddleCo wants to provide its employees with high-speed Internet access. MiddleCo buys a DSL package from Phone East, which, in turn, contracts with service provider Far ISP. Far ISP offers Internet and other IP services to customers who buy DSL from Phone East.

Example of a PPPoE Client Configuration

MiddleCo buys a package from Phone East that provides one DSL line for the site. The package includes a dedicated, authenticated connection to the ISP for MiddleCo's PPPoE clients. The system administrator cables the prospective PPPoE clients to a hub. Technicians from Phone East cable the hub to their DSL equipment.

Example of a PPPoE Server Configuration

To implement the business arrangement FarISP has with Phone East, the system administrator at FarISP configures the access server `dslserve`. This server has the following four interfaces:

- `eri0` – Primary network interface that connects to the local network
- `hme0` – Interface through which FarISP provides Internet service for its customers
- `hme1` – Interface contracted by MiddleCo for authenticated PPPoE tunnels
- `hme2` – Interface contracted by other customers for their PPPoE tunnels

Where to Get More Information About PPPoE

Choose from the following:

- See "Setting Up the PPPoE Client" on page 86.
- See "Setting Up a PPPoE Access Server" on page 88.
- See "Creating PPPoE Tunnels for DSL Support" on page 139, and the `pppoed`(1M), `pppoec`(1M), and `sppptun`(1M) man pages.

◆ ◆ ◆ CHAPTER 3

Setting Up a Dial-up PPP Link (Tasks)

This chapter explains the tasks for configuring the most common PPP link, the dial-up link. Major topics include the following:

Major Tasks for Setting Up the Dial-up PPP Link (Task Map)

You set up the dial-up PPP link by configuring modems, modifying network database files, and modifying the PPP configuration files that are described in Table 8–1.

The next table lists the major tasks to configure both sides of a dial-up PPP link. Typically, you configure only one end of the link, either the dial-out machine or dial-in server.

TABLE 3–1 Task Map for Setting Up the Dial-up PPP Link

Task	Description	For Instructions
1. Gather preconfiguration information	Gather data that is needed prior to setting up the link, such as peer host names, target phone numbers, and modem speed.	"Planning a Dial-up PPP Link" on page 36
2. Configure the dial-out machine	Set up PPP on the machine that makes the call over the link.	"Tasks for Configuring the Dial-out Machine (Task Map)" on page 52
3. Configure the dial-in server	Set up PPP on the machine that receives incoming calls.	"Tasks for Configuring the Dial-in Server (Task Map)" on page 58
4. Call the dial-in server	Type the `pppd` command to initiate communications.	"How to Call the Dial-in Server" on page 62

Configuring the Dial-out Machine

The tasks in this section explain how to configure a dial-out machine. The tasks use as an example the dial- in-from-home scenario that was introduced in Figure 2–1. You can perform the tasks at your company before passing on the machine to a prospective user. Alternatively, you can instruct experienced users in the setup of their home machines. Anyone setting up a dial-out machine must have root permission for that machine.

Tasks for Configuring the Dial-out Machine (Task Map)

TABLE 3–2 Task Map for Setting Up the Dial-out Machine

Task	Description	For Instructions
1. Gather preconfiguration information	Gather data that is needed prior to setting up the link, such as peer host names, target phone numbers, and modem speed.	"Planning a Dial-up PPP Link" on page 36
2. Configure the modem and serial port	Set up the modem and serial port.	"How to Configure the Modem and Serial Port (Dial-out Machine)" on page 53
3. Configure the serial-line communication	Configure the characteristics of the transmission across the serial line.	"How to Define Communications Over the Serial Line" on page 54
4. Define the conversation between the dial-out machine and the peer	Gather communications data for use when you create the chat script.	"How to Create the Instructions for Calling a Peer" on page 55
5. Configure information about a particular peer	Configure PPP options to call an individual dial-in server.	"How to Define the Connection With an Individual Peer" on page 56
6. Call the peer	Type the `pppd` command to initiate communications.	"How to Call the Dial-in Server" on page 62

Dial-up PPP Template Files

Solaris PPP 4.0 provides template files. Each template contains common options for a particular PPP configuration file. The next table lists the sample templates that can be used for setting up a dial-up link, and their equivalent Solaris PPP 4.0 files.

Template File	PPP Configuration File	For Instructions
`/etc/ppp/options.tmpl`	`/etc/ppp/options`	"`/etc/ppp/options.tmpl` Template" on page 116

Template File	PPP Configuration File	For Instructions
`/etc/ppp/options.ttya.tmpl`	`/etc/ppp/options.`*ttyname*	"`options.ttya.tmpl` Template File" on page 118
`/etc/ppp/myisp-chat.tmpl`	File with the name of your choice to contain the chat script	"`/etc/ppp/myisp-chat.tmpl` Chat Script Template" on page 124
`/etc/ppp/peers/myisp.tmpl`	`/etc/ppp/peers/`*peer-name*	"`/etc/ppp/peers/myisp.tmpl` Template File" on page 121

If you decide to use one of the template files, be sure to rename the template to its equivalent PPP configuration file. The one exception is the chat file template `/etc/ppp/myisp-chat.tmpl`. You can choose any name for your chat script.

Configuring Devices on the Dial-out Machine

The first task for setting up a dial-out PPP machine is to configure the devices on the serial line: the modem and serial port.

Note – Tasks that apply to a modem usually apply to an ISDN TA.

Before performing the next procedure, you must have done the following.

- Installed the Oracle Solaris release on the dial-out machine
- Determined the optimum modem speed
- Decided which serial port to use on the dial-out machine
- Obtained the root password for the dial-out machine

For planning information, see "Before You Set Up the Dial-out Machine" on page 36.

▼ How to Configure the Modem and Serial Port (Dial-out Machine)

1 Program the modem.

Even though a variety of modem types is available, most modems are shipped with the correct settings for Solaris PPP 4.0. The following list shows the basic parameter settings for modems that use Solaris PPP 4.0.

- **DCD –** Follow carrier instructions
- **DTR –** Set low so that the modem hangs up and puts the modem on-hook
- **Flow Control –** Set to RTS/CTS for full-duplex hardware flow control

- **Attention Sequences** – Disable

If you have problems setting up the link and suspect that the modem is at fault, first consult the modem manufacturer's documentation. Also, a number of web sites offer help with modem programming. Finally, you can find some suggestions for clearing modem problems in "How to Diagnose Modem Problems" on page 102.

2 **Attach the modem cables to the serial port on the dial-out machine and to the telephone jack.**

3 **Become an administrator on the dial-out machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

4 **Specify modem direction as dial-out only.**

Configuring Communications on the Dial-out Machine

The procedures in this section show how to configure communications over the serial line of the dial-out machine. Before you can use these procedures, you must have configured the modem and serial port, as described in "How to Configure the Modem and Serial Port (Dial-out Machine)" on page 53.

The next tasks show how to enable the dial-out machine to successfully initiate communications with the dial-in server. Communications are initiated as defined in the options in the PPP configuration files. You need to create the following files:

- `/etc/ppp/options`
- `/etc/ppp/options.`*ttyname*
- Chat script
- `/etc/ppp/peers/`*peer-name*

Solaris PPP 4.0 provides templates for the PPP configuration files, which you can customize to accommodate your needs. Refer to "Dial-up PPP Template Files" on page 52 for detailed information about these files.

▼ How to Define Communications Over the Serial Line

1 **Become an administrator on the dial-out machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Create a file that is called `/etc/ppp/options` with the following entry:**

```
lock
```

The `/etc/ppp/options` file is used for defining global parameters that apply to all communications by the local machine. The `lock` option enables UUCP-style locking of the form `/var/spool/locks/LK.xxx.yyy.zzz`.

Note – If the dial-out machine does not have an `/etc/ppp/options` file, only the superuser can run the `pppd` command. However, the `/etc/ppp/options` can be empty.

For a complete description of `/etc/ppp/options`, refer to "`/etc/ppp/options` Configuration File" on page 115.

3 **(Optional) Create a file that is called `/etc/ppp/options.`*ttyname* for defining how communications should be initiated from a specific serial port.**

The next example shows an `/etc/ppp/options.`*ttyname* file for the port with the device name `/dev/cua/a`.

```
# cat /etc/ppp/options.cua.a
crtscts
```

The PPP option `crtscts` tells the `pppd` daemon to turn on hardware flow control for serial port a.

For more information about the `/etc/ppp/options.ttyname` file, go to "`/etc/ppp/options.`*ttyname* Configuration File" on page 116.

4 **Set the modem speed, as described in "How to Set the Modem Speed" on page 59.**

▼ How to Create the Instructions for Calling a Peer

Before the dial-out machine can initiate a PPP link, you must collect information about the dial-in server that is to become the peer. Then, you use this information to create the chat script, which describes the actual conversation between the dial-out machine and the peer.

1 **Determine the speed at which the dial-out machine's modem needs to run.**

For more information, see "Configuring Modem Speed for a Dial-up Link" on page 122.

2 **Obtain the following information from the dial-in server's site.**

- Server's telephone number
- Authentication protocol that is used, if appropriate
- Login sequence that is required by the peer for the chat script

3 **Obtain the names and IP addresses of name servers at the dial-in server's site.**

4 **In a chat script, provide instructions for initiating calls to the particular peer.**

For example, you might create the following chat script, `/etc/ppp/mychat`, to call the dial-in server `myserver`.

```
SAY "Calling the peer\n"
        TIMEOUT 10
        ABORT BUSY
        ABORT 'NO CARRIER'
        ABORT ERROR
        REPORT CONNECT
        "" AT&F1&M5S2=255
        TIMEOUT 60
        OK ATDT1-123-555-1234
        CONNECT \c
        SAY "Connected; logging in.\n"
        TIMEOUT 5
        ogin:--ogin: pppuser
        TIMEOUT 20
        ABORT 'ogin incorrect'
        ssword: \qmypassword
        "% " \c
        SAY "Logged in.  Starting PPP on peer system.\n"
        ABORT 'not found'
        "" "exec pppd"
        ~ \c
```

The script contains instructions for calling a Oracle Solaris dial-in server that requires a login sequence. For a description of each instruction, refer to "Basic Chat Script Enhanced for a UNIX-Style Login" on page 126. For complete details about creating a chat script, read the section "Defining the Conversation on the Dial-up Link" on page 122.

Note – You do not invoke the chat script directly. Rather, you use the file name of the chat script as an argument to the `chat` command, which invokes the script.

If a peer runs Oracle Solaris or a similar operating system, consider using the previous chat script as a template for your dial-out machines.

▼ How to Define the Connection With an Individual Peer

1 **Become an administrator on the dial-out machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Update the repository information for the DNS and name service switch services.**

```
# svccfg
svc:> select network/dns/client
svc:/network/dns/client> setprop config/domain = astring: "bigcompany.com"
svc:/network/dns/client> setprop config/nameserver = net_address: "10.10.111.15"
svc:/network/dns/client> addpropval config/nameserver "10.10.130.8"
```

```
svc:/network/dns/client> select network/dns/client:default
svc:/network/dns/client:default > refresh
svc:/network/dns/client:default > validate
svc:/network/dns/client:default > select system/name-service/switch
svc:/system/name-service/switch > setprop config/host = astring: "files dns"
 svc:/system/name-service/switch:default > select system/name-service/switch:default
svc:/system/name-service/switch:default > refresh
svc:/system/name-service/switch:default > validate
# svcadm enable network/dns/client
# svcadm refresh system/name-service/switch
```

3 Create a file for the peer.

For example, you would create the following file to define the dial-in server `myserver`:

```
# cat /etc/ppp/peers/myserver
/dev/cua/a
57600
noipdefault
defaultroute
idle 120
noauth
connect "chat -U 'mypassword' -T 1-123-555-1213 -f /etc/ppp/mychat"
```

/dev/cua/a
: Specifies that the device `/dev/cua/a` should be used as the serial interface for calls to `myserver`.

57600
: Defines the speed of the link.

noipdefault
: Specifies that for transactions with peer `myserver`, the dial-out machine initially has an IP address of 0.0.0.0. `myserver` assigns an IP address to the dial-out machine for every dial-up session.

idle 120
: Indicates that the link must time out after an idle period of 120 seconds.

noauth
: Specifies that the peer `myserver` does not need to provide authentication credentials when negotiating the connection with the dial-out machine.

connect "chat -U 'mypassword' -T 1-123-555-1213 -f /etc/ppp/mychat"
: Specifies the `connect` option and its arguments, including the phone number of the peer, and the chat script `/etc/ppp/mychat` with calling instructions.

See Also The following list provides references to related information.

- To configure another dial-out machine, see "How to Configure the Modem and Serial Port (Dial-out Machine)" on page 53.
- To test modem connectivity by dialing out to another computer, see `cu(1C)` and `tip(1)` man pages. These utilities can help you test if your modem is properly configured. Also, use these utilities to test if you can establish a connection with another machine.

- To learn more about the configuration files and options, see "Using PPP Options in Files and on the Command Line" on page 111.
- To configure a dial-in server, see "Configuring Devices on the Dial-in Server" on page 58.

Configuring the Dial-in Server

The tasks in this section are for configuring the dial-in server. The dial-in server is a peer machine that receives the call over the PPP link from the dial-out machine. The tasks show how to configure the dial-in server `myserver` that was introduced in Figure 2–1.

Tasks for Configuring the Dial-in Server (Task Map)

TABLE 3–3 Task Map for Setting Up the Dial-in Server

Task	Description	For Instructions
1. Gather preconfiguration information	Gather data that is needed prior to setting up the link, such as peer host names, target phone numbers, and modem speed.	"Planning a Dial-up PPP Link" on page 36
2. Configure the modem and serial port	Set up the modem and serial port.	"How to Configure the Modem and Serial Port (Dial-in Server)" on page 59
3. Configure calling peer information	Set up the user environments and PPP options for every dial-out machine that is permitted to call the dial-in server.	"How to Configure Users of the Dial-in Server" on page 60
4. Configure the serial-line communication	Configure the characteristics of the transmission across the serial line.	"How to Define Communications Over the Serial Line (Dial-in Server)" on page 61

Configuring Devices on the Dial-in Server

The following procedure explains how to configure the modem and serial port on the dial-in server.

Before you do the next procedure, you must have completed the following activities on the peer dial-in server:

- Installed the Oracle Solaris release
- Determined the optimum modem speed
- Decided which serial port to use

How to Configure the Modem and Serial Port (Dial-in Server)

1. **Program the modem, as instructed in the modem manufacturer's documentation.**

   For other suggestions, refer to "How to Configure the Modem and Serial Port (Dial-out Machine)" on page 53.

2. **Attach the modem to the serial port on the dial-in server.**

3. **Become an administrator on the dial-in server.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

4. **Specify modem direction as dial-in only.**

How to Set the Modem Speed

The next procedure explains how to set the modem speed for a dial-in server. For suggestions about speeds to use with Sun Microsystems' computers, see "Configuring Modem Speed for a Dial-up Link" on page 122.

1. **Log in to the dial-in server.**

2. **Use the `tip` command to reach the modem.**

   Instructions for using `tip` to set the modem speed are in the `tip`(1) man page.

3. **Configure the modem for a fixed DTE rate.**

4. **Lock the serial port to that rate, using `ttymon`.**

See Also The following list provides references to related information.

- "How to Configure the Modem and Serial Port (Dial-in Server)" on page 59
- "How to Configure Users of the Dial-in Server" on page 60

Setting Up Users of the Dial-in Server

Part of the process of setting up a dial-in server involves configuring information about each known remote caller.

Before starting the procedures in this section, you must have done the following:

- Obtained the UNIX user names for all users who are permitted to log in from remote dial-out machines.
- Set up the modem and serial line, as described in "How to Configure the Modem and Serial Port (Dial-in Server)" on page 59.
- Dedicated an IP address to be assigned to incoming calls from remote users. Consider creating a dedicated incoming IP address if the number of potential callers exceeds the number of modems and serial ports on the dial-in server. For complete information about creating dedicated IP addresses, go to "Creating an IP Addressing Scheme for Callers" on page 137.

▼ How to Configure Users of the Dial-in Server

1 **Become an administrator on the dial-in server.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Create a new account on the dial-in server for each remote PPP user.**

For instructions about creating a new user, see "Setting Up and Managing User Accounts by Using the CLI (Task Map)" in *Managing User Accounts and User Environments in Oracle Solaris 11.1*.

3 **Create for each caller a `$HOME/.ppprc` file that contains various options that are specific to the user's PPP session.**

For example, you might create the following `.ppprc` file for `pppuser`.

```
# cat /export/home/pppuser/.ppprc
noccp
```

`noccp` turns off compression control on the link.

See Also The following list provides references to related information.

- "How to Configure Users of the Dial-in Server" on page 60.
- "How to Define Communications Over the Serial Line (Dial-in Server)" on page 61.

Configuring Communications Over the Dial-in Server

The next task shows how to enable the dial-in server to open communications with any dial-out machine. The options that are defined in the following PPP configuration files determine how communications are established.

- `/etc/ppp/options`

- `/etc/ppp/options.`*ttyname*

For detailed information about these files, refer to "Using PPP Options in Files and on the Command Line" on page 111.

Before you proceed, you should have done the following:

- Configured the serial port and modem on the dial-in server, as described in "How to Configure the Modem and Serial Port (Dial-in Server)" on page 59.
- Configured information about the prospective users of the dial-in server, as described in "How to Configure Users of the Dial-in Server" on page 60.

▼ How to Define Communications Over the Serial Line (Dial-in Server)

1 Become an administrator on the dial-in server.

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 Create the `/etc/ppp/options` file with the following entry.

```
nodefaultroute
```

`nodefaultroute` indicates that no `pppd` session on the local system can establish a default route without `root` privileges.

Note – If the dial-in server does not have an `/etc/ppp/options` file, only the superuser can run the `pppd` command. However, the `/etc/ppp/options` file can be empty.

---

3 Create the file `/etc/options.`*ttyname* to define how calls that are received over serial port *ttyname* should be handled.

The following `/etc/options.ttya` file defines how the dial-in server's serial port `/dev/ttya` should handle incoming calls.

```
:10.0.0.80
xonxoff
```

`:10.0.0.80`	Assigns the IP address 10.0.0.80 to all peers that are calling in over serial port `ttya`
`xonxoff`	Allows the serial line to handle communications from modems with software flow control enabled

See Also If you have followed all the procedures in this chapter, you have completed the configuration of the dial-up link. The following list provides references to related information.

- To test modem connectivity by dialing out to another computer, see cu(1C) and tip(1) man pages. These utilities can help you test if your modem is properly configured. Also, use these utilities to test if you can establish a connection with another machine.
- To configure more options for the dial-in server, see "Configuring the Dial-in Server" on page 58.
- To configure more dial-out machines, see "Configuring the Dial-out Machine" on page 52.
- To have the remote machine call the dial-in server, see "Calling the Dial-in Server" on page 62.

Calling the Dial-in Server

You establish a dial-up PPP link by having the dial-out machine call the dial-in server. You can instruct the dial-out machine to call the server by specifying the `demand` option in the local PPP configuration files. However, the most common method for establishing the link is for the user to run the `pppd` command on the dial-out machine.

Before you proceed to the next task, you should have done either or both of the following:

- Set up the dial-out machine, as described in "Configuring the Dial-out Machine" on page 52
- Set up the dial-in server, as described in "Configuring the Dial-in Server" on page 58

▼ How to Call the Dial-in Server

1 Log in to the dial-out machine by using your regular user account, not `root`.

2 Call the dial-in server by running the `pppd` command.

For example, the following command initiates a link between the dial-out machine and dial-in server `myserver`:

```
% pppd 57600 call myserver
```

`pppd`	Starts the call by invoking the `pppd` daemon
`57600`	Sets the speed of the line between host and modem
`call myserver`	Invokes the `call` option of `pppd`. `pppd` then reads options in the file `/etc/ppp/peers/myserver`, which was created in "How to Define the Connection With an Individual Peer" on page 56

3 Contact a host on the server's network, for example, the host `lindyhop` that is shown in Figure 2–1:

```
ping lindyhop
```

If the link is not working correctly, refer to Chapter 7, "Fixing Common PPP Problems (Tasks)."

4 **Terminate the PPP session:**

```
% pkill -x pppd
```

See Also If you have followed all the procedures in this chapter, you have completed the configuration of the dial-up link. The following list provides references to related information.

- To have users start working on their dial-out machines, see "How to Call the Dial-in Server" on page 62.
- To fix problems on the link, see Chapter 7, "Fixing Common PPP Problems (Tasks)."
- To learn more about the files and options that are used in this chapter, see "Using PPP Options in Files and on the Command Line" on page 111.

◆ ◆ ◆ CHAPTER 4

Setting Up a Leased-Line PPP Link (Tasks)

This chapter explains how to configure a PPP link that uses a leased line between peers. Major sections include the following:

Setting Up a Leased Line (Task Map)

Leased-line links are relatively easy to set up, in comparison with dial-up links. In most instances, you do not have to configure the CSU/DSU, dialing services, or authentication. If you do need to configure the CSU/DSU, refer to the manufacturer's documentation for aid with this complex task.

The task map in the next table describes all the tasks that are involved in setting up the basic leased-line link.

Note – Some types of leased lines do require the CSU/DSU to "dial" the address of the opposite peer. For example, Frame Relay uses Switched Virtual Circuits (SVCs) or Switched 56 service.

TABLE 4–1 Task Map for Setting Up the Leased-Line Link

Task	Description	For Instructions
1. Gather pre-configuration information	Gather data that is needed prior to setting up the link.	"Information to Be Gathered for the Leased-Line Link" on page 40
2. Set up the leased-line hardware	Assemble the CSU/DSU and synchronous interface card.	"How to Configure Synchronous Devices" on page 66
3. Configure the interface card, if required	Configure the interface script to be used when the leased line is initiated.	"How to Configure Synchronous Devices" on page 66

TABLE 4–1 Task Map for Setting Up the Leased-Line Link *(Continued)*

Task	Description	For Instructions
4. Configure information about the remote peer	Define how communications between your local machine and the remote peer should work.	"How to Configure a Machine on a Leased Line" on page 67
5. Start up the leased line	Configure your machine to start up PPP over the leased line as part of the booting process.	"How to Configure a Machine on a Leased Line" on page 67

Configuring Synchronous Devices on the Leased Line

The task in this section involves configuring equipment that is required by the leased-line topology that is introduced in "Example of a Configuration for a Leased-Line Link" on page 40. The synchronous devices that are required to connect to the leased line include the interface and modem.

Prerequisites for Synchronous Devices Setup

Before you perform the next procedure, you must have the following items:

- Working leased line that is installed at your site by the provider
- Synchronous unit (CSU/DSU)
- Oracle Solaris release installed on your system
- Synchronous interface card of the type that is required by your system

▼ How to Configure Synchronous Devices

1 **Physically install the interface card into the local machine, if necessary.**

Follow the instructions in the manufacturer's documentation.

2 **Connect the cables from the CSU/DSU to the interface.**

If necessary, connect cables from the CSU/DSU to the leased-line jack or similar connector.

3 **Configure the CSU/DSU, as instructed in the documentation from the manufacturer or network provider.**

Note – The provider from whom you rented the leased line might supply and configure the CSU/DSU for your link.

---

4 **Configure the interface card, if necessary, as instructed in the interface documentation.**

The configuration of the interface card involves the creation of a startup script for the interface. The router at LocalCorp in the leased-line configuration that is shown in Figure 2–2 uses an HSI/P interface card.

The following script, `hsi-conf`, starts the HSI/P interface.

```
#!/bin/ksh
/opt/SUNWconn/bin/hsip_init hihp1 speed=1536000 mode=fdx loopback=no \
nrzi=no txc=txc rxc=rxc txd=txd rxd=rxd signal=no 2>&1 > /dev/null
```

`hihp1`	Indicates that HSI/P is the synchronous port used
`speed=1536000`	Set to indicate the speed of the CSU/DSU

See Also To configure the local machine on the leased line, refer to "How to Configure a Machine on a Leased Line" on page 67.

Configuring a Machine on the Leased Line

The task in this section explains how to set up a router to function as the local peer on your end of a leased line. The task uses the leased line that was introduced in "Example of a Configuration for a Leased-Line Link" on page 40 as an example.

Prerequisites for Configuring the Local Machine on a Leased Line

Before you perform the next procedure, you must have completed the following:

- Set up and configure the synchronous devices for the link, as described in "Configuring Synchronous Devices on the Leased Line" on page 66
- Obtained the root password for the local machine on the leased line
- Set up the local machine to run as a router on the network or networks to use the services of the leased-line provider

▼ How to Configure a Machine on a Leased Line

1 **Become an administrator on the local machine (router).**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 Add an entry for the remote peer in the router's `/etc/hosts` file.

```
# cat /etc/hosts
#
# Internet host table
#
127.0.0.1       localhost
192.168.130.10  local2-peer        loghost
192.168.130.11  local1-net
10.0.0.25       farISP
```

The example `/etc/hosts` file is for the local router at the fictitious LocalCorp. Note the IP address and host name for the remote peer `farISP` at the service provider.

3 Create the file `/etc/ppp/peers/`*peer-name* to hold information about the provider's peer.

For this example leased-line link, you create the file `/etc/ppp/peers/farISP`.

```
# cat /etc/ppp/peers/farISP
init '/etc/ppp/conf_hsi'
local
/dev/hihp1
sync
noauth
192.168.130.10:10.0.0.25
passive
persist
noccp
nopcomp
novj
noaccomp
```

The following table explains the options and parameters that are used in `/etc/ppp/peers/farISP`.

Option	Definition
`init '/etc/ppp/conf_hsi'`	Starts the link. `init` then configures the HSI interface by using the parameters in the script `/etc/ppp/conf_hsi`.
`local`	Tells the `pppd` daemon not to change the state of the Data Terminal Ready (DTR) signal. Also tells `pppd` to ignore the Data Carrier Detect (DCD) input signal.
`/dev/hihp1`	Gives the device name of synchronous interface.
`sync`	Establishes synchronous encoding for the link.
`noauth`	Establishes that the local system does not need to demand authentication from the peer. However, the peer could still demand authentication.
`192.168.130.10:10.0.0.25`	Defines the IP addresses of the local peer and the remote peer, separated by a colon.
`passive`	Tells the `pppd` daemon on the local machine to go quiet after issuing maximum number of LCP Configure-Requests and to wait for the peer to start.

Option	Definition
`persist`	Tells the `pppd` daemon to try to restart the link after a connection ends.
`noccp, nopcomp, novj, noaccomp`	Disables the Compression Control Protocol (CCP), Protocol Field compression, Van Jacobson compression, and address and control field compression, respectively. These forms of compression accelerate transmissions on a dial-up link but could slow down a leased line.

4 Create an initialization script that is called `demand`, which creates the PPP link as part of the booting process.

```
# cat /etc/ppp/demand
#!/bin/sh
if [ -f /system/volatile/ppp-demand.pid ] &&
   /usr/bin/kill -s 0 ‘/bin/cat /system/volatile/ppp-demand.pid‘
then
        :
else
        /usr/bin/pppd call farISP
fi
```

The `demand` script contains the `pppd` command for establishing a leased-line link. The following table explains the content of `$PPPDIR/demand`.

Code Sample	Explanation
`if [ -f /system/volatile/ppp-demand.pid ] && /usr/bin/kill -s 0 ‘/bin/cat /system/volatile/ppp-demand.pid‘`	These lines check to see if `pppd` is running. If `pppd` is running, it does not need to be started.
`/usr/bin/pppd call farISP`	This line launches `pppd`. `pppd` reads the options from `/etc/ppp/options`. The `call farISP` option on the command line causes it to read `/etc/ppp/peers/farISP`, also.

The Solaris PPP 4.0 startup script `/etc/rc2.d/S47pppd` invokes the `demand` script as part of the booting process. The following lines in `/etc/rc2.dS47pppd` search for the presence of a file that is called `$PPPDIR/demand`.

```
if [ -f $PPPDIR/demand ]; then
            . $PPPDIR/demand
    fi
```

If found, `$PPPDIR/demand` is executed. During the course of executing `$PPPDIR/demand`, the link is established.

Note – To reach machines outside the local network, have users run `telnet`, `ftp`, `rsh`, or similar commands.

See Also If you have followed all the procedures in this chapter, you have completed the configuration of the leased-line link. The following list provides references to related information.

- To find troubleshooting information, see "Fixing Leased-Line Problems" on page 108.
- To learn more about the files and options that are used in this chapter, see "Using PPP Options in Files and on the Command Line" on page 111.

◆ ◆ ◆ CHAPTER 5

Setting Up PPP Authentication (Tasks)

This chapter contains tasks for setting up PPP authentication. Subjects that are covered include the following:

- "Configuring PAP Authentication" on page 72
- "Configuring CHAP Authentication" on page 79

The procedures show how to implement authentication over a dial-up link because dial-up links are more likely to be configured for authentication than leased-line links. You can configure authentication over leased lines if authentication is required by your corporate security policy. For leased-line authentication, use the tasks in this chapter as guidelines.

If you want to use PPP authentication but are not sure which protocol to use, review the section "Why Use PPP Authentication?" on page 30. More detailed information about PPP authentication is in the `pppd`(1M) man page and in "Authenticating Callers on a Link" on page 131.

Configuring PPP Authentication (Task Map)

This section contains task maps to help you quickly access procedures for PPP authentication.

TABLE 5–1 Task Map for General PPP Authentication

Task	Description	For Instructions
Configure PAP authentication	Use these procedures to enable PAP authentication on a dial-in server and a dial-out machine.	"Setting Up PAP Authentication (Task Maps)" on page 72
Configure CHAP authentication	Use these procedures to enable CHAP authentication on a dial-in server and a dial-out machine.	"Setting Up CHAP Authentication (Task Maps)" on page 79

Configuring PAP Authentication

The tasks in this section explain how to implement authentication on a PPP link by using the Password Authentication Protocol (PAP). The tasks use the example that is shown in "Examples of PPP Authentication Configurations" on page 42 to illustrate a working PAP scenario for a dial-up link. Use the instructions as the basis for implementing PAP authentication at your site.

Before you perform the next procedures, you must have done the following:

- Set up and tested the dial-up link between the dial-in server and dial-out machines that belong to trusted callers
- Ideally, for dial-in server authentication, obtained superuser permission for the machine where the network password database is administered, for example, in LDAP, NIS, or local files
- Obtained superuser authority for the local machine, either dial-in server or dial-out machine

Setting Up PAP Authentication (Task Maps)

Use the next task maps to quickly access PAP-related tasks for the dial-in server and trusted callers on dial-out machines.

TABLE 5–2 Task Map for PAP Authentication (Dial-in Server)

Task	Description	For Instructions
1. Gather preconfiguration information	Collect user names and other data that is needed for authentication.	"Planning for Authentication on a Link" on page 41
2. Update the password database, if necessary	Ensure that all potential callers are in the server's password database.	"How to Create a PAP Credentials Database (Dial-in Server)" on page 73
3. Create the PAP database	Create security credentials for all prospective callers in `/etc/ppp/pap-secrets`.	"How to Create a PAP Credentials Database (Dial-in Server)" on page 73
4. Modify the PPP configuration files	Add options specific to PAP to the `/etc/ppp/options` and `/etc/ppp/peers/`*peer-name* files.	"How to Add PAP Support to the PPP Configuration Files (Dial-in Server)" on page 75

TABLE 5–3 Task Map for PAP Authentication (Dial-out Machine)

Task	Description	For Instructions
1. Gather preconfiguration information	Collect user names and other data that is needed for authentication.	"Planning for Authentication on a Link" on page 41

TABLE 5–3 Task Map for PAP Authentication (Dial-out Machine) *(Continued)*

Task	Description	For Instructions
2. Create the PAP database for the trusted caller's machine	Create the security credentials for the trusted caller and, if necessary, security credentials for other users who call the dial-out machine, in `/etc/ppp/pap-secrets`.	"How to Configure PAP Authentication Credentials for the Trusted Callers" on page 76
3. Modify the PPP configuration files	Add options specific to PAP to the `/etc/ppp/options` and `/etc/ppp/peers/`*peer-name* files.	"How to Add PAP Support to the PPP Configuration Files (Dial-out Machine)" on page 78

Configuring PAP Authentication on the Dial-in Server

To set up PAP authentication, you must do the following:

- Create a PAP credentials database
- Modify PPP configuration files for PAP support

▼ How to Create a PAP Credentials Database (Dial-in Server)

This procedure modifies the `/etc/ppp/pap-secrets` file, which contains the PAP security credentials that are used to authenticate callers on the link. `/etc/ppp/pap-secrets` must exist on both machines on a PPP link.

The sample PAP configuration that was introduced in Figure 2–3 uses the `login` option of PAP. If you plan to use this option, you might also need to update your network's password database. For more information about the `login` option, refer to "Using the `login` Option With `/etc/ppp/pap-secrets`" on page 134.

1 **Assemble a list of all potential trusted callers. Trusted callers are people to be granted permission to call the dial-in server from their remote machines.**

2 **Verify that each trusted caller already has a UNIX user name and password in the dial-in server's password database.**

Note – Verification is particularly important for the sample PAP configuration, which uses the `login` option of PAP to authenticate callers. If you choose not to implement `login` for PAP, the callers' PAP user names do not have to correspond with their UNIX user names. For information about standard `/etc/ppp/pap-secrets`, refer to "`/etc/ppp/pap-secrets` File" on page 132.

Do the following if a potential trusted caller does not have a UNIX user name and password:

a. **Confirm with their managers that callers whom you do not know personally have permission to access the dial-in server.**

b. **Create UNIX user names and passwords for these callers in the manner that is directed by your corporate security policy.**

3 **Become an administrator on the dial-in server.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

4 **Edit the `/etc/ppp/pap-secrets` file.**

This release provides a `pap-secrets` file in `/etc/ppp` that contains comments about how to use PAP authentication but no options. You can add the following options at the end of the comments.

```
user1      myserver        ""          *
user2      myserver        ""          *
myserver   user2           serverpass  *
```

To use the `login` option of `/etc/ppp/pap-secrets`, you must type the UNIX user name of each trusted caller. Wherever a set of double quotes ("") appears in the third field, the password for the caller is looked up in the server's password database.

The entry `myserver * serverpass *` contains the PAP user name and password for the dial-in server. In Figure 2–3, the trusted caller `user2` requires authentication from remote peers. Therefore, `myserver`'s `/etc/ppp/pap-secrets` file contains PAP credentials for use when a link is established with `user2`.

See Also The following list provides references to related information.

- "Modifying the PPP Configuration Files for PAP (Dial-in Server)" on page 74
- "Configuring PAP Authentication for Trusted Callers (Dial-out Machines)" on page 76

Modifying the PPP Configuration Files for PAP (Dial-in Server)

The tasks in this section explain how to update any existing PPP configuration files to support PAP authentication on the dial-in server.

How to Add PAP Support to the PPP Configuration Files (Dial-in Server)

The procedure uses as examples the PPP configuration files that were introduced in "How to Define Communications Over the Serial Line (Dial-in Server)" on page 61.

1 Become an administrator on the dial-in server.

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 Add authentication options to the `/etc/ppp/options` file.

For example, you would add the options in bold to an existing `/etc/ppp/options` file to implement PAP authentication:

```
lock
auth
login
nodefaultroute
proxyarp
ms-dns 10.0.0.1
idle 120
```

`auth`	Specifies that the server must authenticate callers before establishing the link.
`login`	Specifies that the remote caller be authenticated by using the standard UNIX user authentication services.
`nodefaultroute`	Indicates that no `pppd` session on the local system can establish a default route without `root` privileges.
`proxyarp`	Adds an entry to the system's Address Resolution Protocol (ARP) table that specifies the IP address of the peer and the Ethernet address of the system. With this option the peer appears to be on the local Ethernet to other systems.
`ms-dns 10.0.0.1`	Enables `pppd` to supply a Domain Name Server (DNS) address, `10.0.0.1`, for the client
`idle 120`	Specifies that idle users are disconnected after two minutes.

3 In the `/etc/ppp/options.cua.a` file, add the following address for the `cua/a` user.

```
:10.0.0.2
```

4 In the `/etc/ppp/options.cua.b` file, add the following address for the `cua/b` user.

```
:10.0.0.3
```

5 **In the `/etc/ppp/pap-secrets` file, add the following entry.**

```
*       *           ""      *
```

Note – The `login` option, as previously described, supplies the necessary user authentication. This entry in the `/etc/ppp/pap-secrets` file is the standard way of enabling PAP with the `login` option.

See Also To configure PAP authentication credentials for trusted callers of the dial-in server, refer to "Configuring PAP Authentication for Trusted Callers (Dial-out Machines)" on page 76.

Configuring PAP Authentication for Trusted Callers (Dial-out Machines)

This section contains tasks for setting up PAP authentication on the dial-out machines of trusted callers. As system administrator, you can set up PAP authentication on the systems before distribution to prospective callers. Or, if the remote callers already have their machines, you can give these callers the tasks in this section.

Configuring PAP for trusted callers involves two tasks:

- Configuring the callers' PAP security credentials
- Configuring the callers' dial-out machines to support PAP authentication

▼ How to Configure PAP Authentication Credentials for the Trusted Callers

This procedure shows how to set up PAP credentials for two trusted callers, one of which requires authentication credentials from remote peers. The steps in the procedure assume that you, the system administrator, are creating the PAP credentials on the trusted callers' dial-out machines.

1 **Become an administrator on the dial-out machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

Using the sample PAP configuration that was introduced in Figure 2–3, assume that the dial-out machine belongs to `user1`.

2 **Modify the pap-secrets database for the caller.**

This release provides an `/etc/ppp/pap-secrets` file that contains helpful comments but no options. You can add the following options to this `/etc/ppp/pap-secrets` file.

```
user1    myserver  pass1    *
```

Note that `user1`'s password `pass1` is passed in readable ASCII form over the link. `myserver` is caller `user1`'s name for the peer.

3 **Become an administrator on the dial-out machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

Using the PAP authentication example, assume that this dial-out machine belongs to the caller `user2`.

4 **Modify the pap-secrets database for the caller.**

You can add the next options to the end of the existing `/etc/ppp/pap-secrets` file.

```
user2     myserver   pass2       *
myserver  user2      serverpass  *
```

In this example, `/etc/ppp/pap-secrets` has two entries. The first entry contains the PAP security credentials that `user2` passes to dial-in server `myserver` for authentication.

`user2` requires PAP credentials from the dial-in server as part of link negotiation. Therefore, the `/etc/ppp/pap-secrets` also contains PAP credentials that are expected from `myserver` on the second line.

Note – Because most ISPs do not supply authentication credentials, the previous scenario might be unrealistic for communications with an ISP.

See Also The following list provides references to related information.

- "How to Create a PAP Credentials Database (Dial-in Server)" on page 73
- "How to Configure PAP Authentication Credentials for the Trusted Callers" on page 76

Modifying PPP Configuration Files for PAP (Dial-out Machine)

The following tasks explain how to update existing PPP configuration files to support PAP authentication on the dial-out machines of trusted callers.

The procedure uses the following parameters to configure PAP authentication on the dial-out machine that belongs to `user2`, who was introduced in Figure 2–3. `user2` requires incoming callers to authenticate, including calls from dial-in `myserver`.

How to Add PAP Support to the PPP Configuration Files (Dial-out Machine)

This procedure uses as examples the PPP configuration files that were introduced in "How to Define Communications Over the Serial Line" on page 54. The procedure configures the dial-out machine that belongs to `user2`, as shown in Figure 2–3.

1. **Log in to the dial-out machine as superuser.**

2. **Modify the `/etc/ppp/options` file.**

   The next `/etc/ppp/options` file contains options for PAP support, which are shown in bold.

   ```
   # cat /etc/ppp/options
   lock
   name user2
   auth
   require-pap
   ```

   name user2 — Sets `user2` as the PAP name of the user on the local machine. If the `login` option is used, the PAP name must be the same as the user's UNIX user name in the password database.

   auth — States that the dial-out machine must authenticate callers before establishing the link.

   Note – This dial-out machine demands authentication from its peers, even though most dial-out machines do not make this demand. Either way is acceptable.

   require-pap — Demands PAP credentials from the peer.

3. **Create an `/etc/ppp/peers/`*peer-name* file for the remote machine `myserver`.**

   The next example shows how to add PAP support to the existing `/etc/ppp/peers/myserver` file that was created in "How to Define the Connection With an Individual Peer" on page 56.

   ```
   # cat /etc/ppp/peers/myserver
   /dev/cua/a
   57600
   noipdefault
   defaultroute
   idle 120
   ```

```
user user2
remotename myserver
connect "chat -U 'mypassword' -f /etc/ppp/mychat"
```

The new options in bold add PAP requirements for peer `myserver`.

user user2	Defines `user2` as the user name of the local machine
remotename myserver	Defines `myserver` as a peer that requires authentication credentials from the local machine

See Also The following list provides references to related information.

- To test the PAP authentication setup by calling the dial-in server, see "How to Call the Dial-in Server" on page 62.
- To learn more about PAP authentication, see "Password Authentication Protocol (PAP)" on page 131.

Configuring CHAP Authentication

The tasks in this section explain how to implement authentication on a PPP link by using the Challenge-Handshake Authentication Protocol (CHAP). The tasks use the example that is shown in Figure 2–4 to illustrate a working CHAP scenario for dialing up a private network. Use the instructions as the basis for implementing CHAP authentication at your site.

Before you perform the next procedures, you must have done the following:

- Set up and tested the dial-up link between the dial-in server and dial-out machines that belong to trusted callers
- Obtained superuser permission for the local machine, either dial-in server or dial-out machine

Setting Up CHAP Authentication (Task Maps)

TABLE 5–4 Task Map for CHAP Authentication (Dial-in Server)

Task	Description	For Instructions
1. Assign CHAP secrets to all trusted callers	Create, or have the callers create, their CHAP secrets.	"How to Create a CHAP Credentials Database (Dial-in Server)" on page 81
2. Create the chap-secrets database	Add the security credentials for all trusted callers to the `/etc/ppp/chap-secrets` file.	"How to Create a CHAP Credentials Database (Dial-in Server)" on page 81

TABLE 5–4 Task Map for CHAP Authentication (Dial-in Server) *(Continued)*

Task	Description	For Instructions
3. Modify the PPP configuration files	Add options specific to CHAP to the `/etc/ppp/options` and `/etc/ppp/peers/`*peer-name* files.	"How to Add CHAP Support to the PPP Configuration Files (Dial-in Server)" on page 82

TABLE 5–5 Task Map for CHAP Authentication (Dial-out Machine)

Task	Description	For Instructions
1. Create the CHAP database for the trusted caller's machine	Create the security credentials for the trusted caller and, if necessary, security credentials for other users who call the dial-out machine, in `/etc/ppp/chap-secrets`.	"How to Create a CHAP Credentials Database (Dial-in Server)" on page 81
2. Modify the PPP configuration files	Add options specific to CHAP to the `/etc/ppp/options` file.	"How to Add CHAP Support to the PPP Configuration Files (Dial-out Machine)" on page 84

Configuring CHAP Authentication on the Dial-in Server

The first task in setting up CHAP authentication is modifying the `/etc/ppp/chap-secrets` file. This file contains the CHAP security credentials, including the CHAP secret, that are used to authenticate callers on the link.

Note – UNIX or PAM authentication mechanisms do not work with CHAP. For example, you cannot use the PPP `login` option as described in "How to Create a PAP Credentials Database (Dial-in Server)" on page 73. If your authentication scenario requires PAM or UNIX-style authentication, choose PAP instead.

The next procedure implements CHAP authentication for a dial-in server in a private network. The PPP link is the only connection to the outside world. The only callers who can access the network have been granted permission by managers of the network, possibly including the system administrator.

▼ How to Create a CHAP Credentials Database (Dial-in Server)

1 **Assemble a list that contains the user names of all trusted callers.**

Trusted callers include all people who have been granted permission to call the private network.

2 **Assign each user a CHAP secret.**

Note – Be sure to choose a good CHAP secret that is not easily guessed. No other restrictions are placed on the CHAP secret's contents.

The method for assigning CHAP secrets depends on your site's security policy. Either you have the responsibility for creating the secrets, or the callers must create their own secrets. If you are not responsible for CHAP secret assignment, be sure to get the CHAP secrets that were created by, or for, each trusted caller.

3 **Become an administrator on the dial-in server.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

4 **Modify the `/etc/ppp/chap-secrets` file.**

This release includes an `/etc/ppp/chap-secrets` file that contains helpful comments but no options. You can add the following options for the server `CallServe` at the end of the existing `/etc/ppp/chap-secrets` file.

```
account1  CallServe   key123   *
account2  CallServe   key456   *
```

`key123` is the CHAP secret for trusted caller `account1`.

`key456` is the CHAP secret for trusted caller `account2`.

See Also The following list provides references to related information.

- "How to Create a CHAP Credentials Database (Dial-in Server)" on page 81
- "How to Add CHAP Support to the PPP Configuration Files (Dial-in Server)" on page 82
- "Configuring CHAP Authentication for Trusted Callers (Dial-out Machines)" on page 82

Modifying the PPP Configuration Files for CHAP (Dial-in Server)

The task in this section explains how to update existing PPP configuration files to support CHAP authentication on the dial-in server.

▼ How to Add CHAP Support to the PPP Configuration Files (Dial-in Server)

1. **Log in to the dial-in server as superuser.**

2. **Modify the `/etc/ppp/options` file.**

   Add the options that are shown in bold for CHAP support.

   ```
   # cat /etc/ppp/options
   lock
   nodefaultroute
   name CallServe
   auth
   ```

`name` *CallServe*	Defines *CallServe* as the CHAP name of the user on the local machine, in this instance the dial-in server
`auth`	Makes the local machine authenticate callers before establishing the link

3. **Create the remaining PPP configuration files to support the trusted callers.**

   See "How to Configure Users of the Dial-in Server" on page 60 and "How to Define Communications Over the Serial Line (Dial-in Server)" on page 61.

See Also To configure CHAP authentication credentials for trusted callers, refer to "How to Create a CHAP Credentials Database (Dial-in Server)" on page 81.

Configuring CHAP Authentication for Trusted Callers (Dial-out Machines)

This section contains tasks for setting up CHAP authentication on the dial-out machines of trusted callers. Depending on your site's security policy, either you or the trusted callers might be responsible for setting up CHAP authentication.

For remote callers to configure CHAP, ensure that the callers' local CHAP secrets match the callers' equivalent CHAP secrets in the dial-in server's `/etc/ppp/chap-secrets` file. Then give the callers the tasks in this section for configuring CHAP.

Configuring CHAP for trusted callers involves two tasks:

- Creating the callers' CHAP security credentials
- Configuring the callers' dial-out machines to support CHAP authentication

How to Configure CHAP Authentication Credentials for the Trusted Callers

This procedure shows how to set up CHAP credentials for two trusted callers. The steps in the procedure assume that you, the system administrator, are creating the CHAP credentials on the trusted callers' dial-out machines.

1. **Become an administrator on the dial-out machine.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

   Using the sample CHAP configuration in "Example of a Configuration Using CHAP Authentication" on page 45, assume that the dial-out machine belongs to trusted caller `account1`.

2. **Modify the chap-secrets database for caller `account1`.**

   This release includes an `/etc/ppp/chap-secrets` file that has helpful comments but no options. You can add the following options to the existing `/etc/ppp/chap-secrets` file.

   ```
   account1  CallServe   key123   *
   ```

   `CallServe` is the name for the peer that `account1` is trying to reach. `key123` is the CHAP secret to be used for links between `account1` and `CallServer`.

3. **Become an administrator on the dial-out machine.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

   Assume that this machine belongs to caller `account2`.

4. **Modify the `/etc/ppp/chap-secrets` database for caller account2.**

   ```
   account2  CallServe   key456   *
   ```

   Now, `account2` has secret `key456` as its CHAP credentials for use over links to peer `CallServe`.

See Also The following list provides references to related information.

- "How to Create a CHAP Credentials Database (Dial-in Server)" on page 81
- "How to Configure CHAP Authentication Credentials for the Trusted Callers" on page 83

Adding CHAP to the Configuration Files (Dial-out Machine)

To learn more about CHAP authentication, refer to "Challenge-Handshake Authentication Protocol (CHAP)" on page 134. The next task configures the dial-out machine that belongs to caller `account1`, which is introduced in "Example of a Configuration Using CHAP Authentication" on page 45.

▼ How to Add CHAP Support to the PPP Configuration Files (Dial-out Machine)

1. **Log in to the dial-out machine as superuser.**

2. **Ensure that the `/etc/ppp/options` file has the following options.**

```
# cat /etc/ppp/options
lock
nodefaultroute
```

3. **Create an `/etc/ppp/peers/`*peer-name* file for the remote machine `CallServe`.**

```
# cat /etc/ppp/peers/CallServe
/dev/cua/a
57600
noipdefault
defaultroute
idle 120
user account1
connect "chat -U 'mypassword' -f /etc/ppp/mychat"
```

The option `user account1` sets `account1` as the CHAP user name to be given to `CallServe`. For a description of the other options in the previous file, see the similar `/etc/ppp/peers/myserver` file in "How to Define the Connection With an Individual Peer" on page 56.

See Also To test CHAP authentication by calling the dial-in server, refer to "How to Call the Dial-in Server" on page 62.

◆ ◆ ◆ CHAPTER 6

Setting Up a PPPoE Tunnel (Tasks)

This chapter contains tasks for setting up the participants on either end of the PPPoE tunnel: the PPPoE client and PPPoE access server. Specific topics include the following:

- "Major Tasks for Setting Up a PPPoE Tunnel (Task Maps)" on page 85
- "Setting Up the PPPoE Client" on page 86
- "Setting Up a PPPoE Access Server" on page 88

The tasks use the scenario that was introduced in "Planning for DSL Support Over a PPPoE Tunnel" on page 47 as an example. For an overview of PPPoE, refer to "Support for DSL Users Through PPPoE" on page 31.

Major Tasks for Setting Up a PPPoE Tunnel (Task Maps)

The following tables list the major tasks for configuring PPPoE clients and the PPPoE access server. To implement PPPoE at your site, you need to set up only your end of the PPPoE tunnel, either the client side or access-server side.

TABLE 6–1 Task Map for Setting Up a PPPoE Client

Task	Description	For Instructions
1. Configure an interface for PPPoE	Define the Ethernet interface to be used for the PPPoE tunnel.	"How to Configure an Interface for a PPPoE Client" on page 86
2. Configure information about the PPPoE access server	Define parameters for the access server at the service provider end of the PPPoE tunnel.	"How to Define a PPPoE Access Server Peer" on page 87
3. Set up the PPP configuration files	Define the PPP configuration files for the client, if you have not done so already.	"How to Define Communications Over the Serial Line" on page 54
4. Create the tunnel	Call the access server.	"How to Define a PPPoE Access Server Peer" on page 87

TABLE 6–2 Task Map for Setting Up a PPPoE Access Server

Task	Description	For Instructions
1. Set up a PPPoE access server	Define the Ethernet interface to be used for the PPPoE tunnel and define the services that the access server offers.	"How to Set Up a PPPoE Access Server" on page 89
2. Set up the PPP configuration files	Define the PPP configuration files for the client, if you have not done so already.	"Configuring Communications Over the Dial-in Server" on page 60
3. (Optional) Restrict use of an interface	Use PPPoE options and PAP authentication to restrict use of a particular Ethernet interface to certain clients.	"How to Restrict the Use of an Interface to Particular Clients" on page 90

Setting Up the PPPoE Client

To provide PPP to client systems over DSL, you must first configure PPPoE on the interface that is connected to the modem or hub. Then you need to change the PPP configuration files to define the access server on the opposite end of the PPPoE.

Prerequisites for Setting Up the PPPoE Client

Before you set up the PPPoE client, you must have done the following:

- Installed Oracle Solaris release on the client machines to use the PPPoE tunnel.
- Contacted the service provider for information about its PPPoE access server.
- Had the telephone company or service provider assemble the devices that are used by the client machines. These devices include, for example, the DSL modem and the splitter, which the telephone company rather than you might assemble.

▼ How to Configure an Interface for a PPPoE Client

Use this procedure to define the Ethernet interface to be used for the PPPoE tunnel.

1 Become an administrator on the PPPoE client.

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 Add the name of the Ethernet interface with the DSL connection to the `/etc/ppp/pppoe.if` file.

For example, you add the following entry to `/etc/ppp/pppoe.if` for a PPPoE client that uses `hme0` as the network interface that is connected to the DSL modem.

```
hme0
```

For more information about `/etc/ppp/pppoe.if`, go to "`/etc/ppp/pppoe.if` File" on page 140.

3 **Configure the interface for PPPoE use.**

```
# /etc/init.d/pppd start
```

4 **(Optional) Verify that the interface is now plumbed for PPPoE.**

```
# /usr/sbin/sppptun query
hme0:pppoe
hme0:pppoed
```

You can also use the `/usr/sbin/sppptun` command to manually plumb interfaces for PPPoE. For instructions, refer to "`/usr/sbin/sppptun` Command" on page 140.

How to Define a PPPoE Access Server Peer

You define the access server in the `/etc/ppp/peers/`*peer-name* file. Many of the options that are used for the access server are also used to define the dial-in server in a dial-up scenario. For a detailed explanation of `/etc/ppp/peers.`*peer-name*, refer to "`/etc/ppp/peers/`*peer-name* File" on page 120.

1 **Become an administrator on the PPPoE client.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Define the service provider's PPPoE access server in the `/etc/ppp/peers/`*peer-name* file.**

For example, the following file, `/etc/ppp/peers/dslserve`, defines the access server `dslserve` at Far ISP that is introduced in "Example of a Configuration for a PPPoE Tunnel" on page 49.

```
# cat /etc/ppp/peers/dslserve
sppptun
plugin pppoe.so
connect "/usr/lib/inet/pppoec hme0"
noccp
noauth
user Red
password redsecret
noipdefault
defaultroute
```

For a definition of the options in this file, go to "`/etc/ppp/peers/`*peer-name* File for Defining an Access Server Peer" on page 147.

3 **Modify the other PPP configuration files on the PPPoE client.**

a. **Configure `/etc/ppp/options` as described in the instructions for configuring a dial-out machine in "Configuring the Dial-out Machine" on page 52.**

b. Create an `/etc/ppp/options.sppptun` file. `/etc/ppp/options.sppptun` defines PPP options for the serial port to which the interface that is plumbed for PPPoE is attached.

You can use any options that are available for the `/etc/ppp/options.`*ttyname* file that is described in "`/etc/ppp/options.`*ttyname* Configuration File" on page 116. You must name the file `/etc/ppp/options.sppptun` because `sppptun` is the specified device name in the `pppd` configuration.

4 Ensure that all users can start PPP on the client.

```
# touch /etc/ppp/options
```

5 Test if PPP can run over the DSL line.

```
% pppd debug updetach call dslserve
```

`dslserve` is the name that is given to the access server at the ISP that is shown in "Example of a Configuration for a PPPoE Tunnel" on page 49. The `debug updetach` option causes debugging information to be displayed in a terminal window.

If PPP is running correctly, the terminal output shows the link becoming active. If PPP still does not run, try the following command to see if the servers are running correctly:

```
# /usr/lib/inet/pppoec -i hme0
```

Note – Users of configured PPPoE clients can begin running PPP over a DSL line by typing the following:

```
% pppd call ISP-server-name
```

Then the users can run an application or a service.

See Also The following list provides references to related information.

- See "Setting Up the PPPoE Client" on page 86.
- See "Creating PPPoE Tunnels for DSL Support" on page 139.
- See Chapter 7, "Fixing Common PPP Problems (Tasks)."
- See "Setting Up a PPPoE Access Server" on page 88.

Setting Up a PPPoE Access Server

If your company is a service provider, you can offer Internet and other services to clients that reach your site through DSL connections. The procedure involves determining which interfaces on the server to involve in the PPPoE tunnel and defining which services are made available to the users.

How to Set Up a PPPoE Access Server

Use this procedure to define the Ethernet interface to be used for the PPPoE tunnel and to configure the services that the access server offers.

1. **Become an administrator on the access server.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Add the name of the Ethernet interfaces that are dedicated to the PPPoE tunnels to the `/etc/ppp/pppoe.if` file.**

   For example, you would use the following `/etc/ppp/pppoe.if` file for the access server `dslserve` that is shown in "Example of a Configuration for a PPPoE Tunnel" on page 49.

   ```
   # cat /etc/ppp/pppoe.if
   hme1
   hme2
   ```

3. **Define global services that are provided by the access server in the `/etc/ppp/pppoe` file.**

   The following `/etc/ppp/pppoe` file lists the services that are provided by access server `dslserve`, which was shown in Figure 2–5.

   ```
   device hme1,hme2
   service internet
       pppd "proxyarp 192.168.1.1:"
   service debugging
       pppd "debug proxyarp 192.168.1.1:"
   ```

   In the file example, Internet service is announced for `dslserve`'s Ethernet interfaces `hme1` and `hme2`. Debugging is turned on for PPP links on the Ethernet interfaces.

4. **Set up the PPP configuration files in the same way that you would for a dial-in server.**

   For more information, refer to "Creating an IP Addressing Scheme for Callers" on page 137.

5. **Start the pppoed daemon.**

   ```
   # /etc/init.d/pppd start
   ```

   `pppd` also plumbs the interfaces that are listed in `/etc/ppp/pppoe.if`.

6. **(Optional) Verify that the interfaces on the server are plumbed for PPPoE.**

   ```
   # /usr/sbin/sppptun query
   hme1:pppoe
   hme1:pppoed
   hme2:pppoe
   hme2:pppoed
   ```

   The previous sample shows that interfaces `hme1` and `hme2` are currently plumbed for PPPoE. You can also use the `/usr/sbin/sppptun` command to manually plumb interfaces for PPPoE. For instructions, refer to "`/usr/sbin/sppptun` Command" on page 140.

How to Modify an Existing /etc/ppp/pppoe File

1. **Become an administrator on the access server.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Modify /etc/ppp/pppoe, as needed.**

3. **Cause the pppoed daemon to recognize the new services.**

   ```
   # pkill -HUP pppoed
   ```

How to Restrict the Use of an Interface to Particular Clients

The next procedure shows how to restrict an interface to a group of PPPoE clients. Before performing this task, you need to obtain the real Ethernet MAC addresses of the clients you are assigning to the interface.

Note – Some systems allow you to change the MAC address on the Ethernet interface. You should view this ability as a convenience factor, not a security measure.

Using the example that is shown in "Example of a Configuration for a PPPoE Tunnel" on page 49, these steps show how to reserve one of dslserve's interfaces, hme1, for clients at MiddleCo.

1. **Configure the access server's interfaces and define the services, as shown in "How to Set Up a PPPoE Access Server" on page 89.**

2. **Create entries for clients in the server's /etc/ethers database.**

   Here is a sample entry for clients Red, Blue, and Yellow.

   ```
   8:0:20:1:40:30 redether
   8:0:20:1:40:10 yellowether
   8:0:20:1:40:25 blueether
   ```

   The sample assigns the symbolic names redether, yellowether, and blueether to the Ethernet addresses of clients Red, Yellow, and Blue. The assignment of symbolic names to the MAC addresses is optional.

3 Restrict services that are provided on a specific interface by defining the following information in the `/etc/ppp/pppoe.`*device* file.

In this file, *device* is the name of the device to be defined.

```
# cat /etc/ppp/pppoe.hme1
service internet
     pppd "name dslserve-hme1"
     clients redether,yellowether,blueether
```

`dslserve-hme1` is the access server's name, which is used in matching entries in the `pap-secrets` file. The `clients` option restricts the use of interface `hme1` to clients with the symbolic Ethernet names `redether`, `yellowether`, and `blueether`.

If you did not define symbolic names for client's MAC addresses in `/etc/ethers`, you can use the numeric addresses as arguments for the `clients` option. Wildcards are allowed.

For example, you can specify the numeric address `clients 8:0:20:*:*:*`. By using wildcards, all matching addresses in `/etc/ethers` are accepted.

4 Create the `/etc/ppp/pap-secrets` file for the access server:

```
Red         dslserve-hme1   redpasswd     *
Blue        dslserve-hme1   bluepasswd    *
Yellow      dslserve-hme1   yellowpassd   *
```

The entries are the PAP names and passwords of clients that are allowed to run PPP over `dslserve`'s `hme1` interface.

For more information about PAP authentication, see "Configuring PAP Authentication" on page 72.

See Also The following list provides references to related information.

- To learn more about PPPoE, see "Creating PPPoE Tunnels for DSL Support" on page 139.
- To troubleshoot PPPoE and PPP problems, see "Solving PPP-Related and PPPoE-Related Problems" on page 97.
- To configure a PPPoE client, see "Setting Up the PPPoE Client" on page 86.
- To configure PAP authentication for a client, see "Configuring PAP Authentication for Trusted Callers (Dial-out Machines)" on page 76.
- To configure PAP authentication on a server, see "Configuring PAP Authentication on the Dial-in Server" on page 73.

◆ ◆ ◆ CHAPTER 7

Fixing Common PPP Problems (Tasks)

This chapter contains information for troubleshooting common problems that occur with Solaris PPP 4.0. The following topics are covered:

- "Tools for Troubleshooting PPP" on page 94
- "Solving PPP-Related and PPPoE-Related Problems" on page 97
- "Fixing Leased-Line Problems" on page 108
- "Diagnosing and Fixing Authentication Problems" on page 109

The sources *PPP Design, Implementation, and Debugging* by James Carlson and the Australian National University's web site also have detailed advice for PPP troubleshooting. For more information, see "Professional Reference Books About PPP" on page 21 and "Web Sites About PPP" on page 22.

Solving PPP Problems (Task Map)

Use the following task map to quickly access advice and solutions for common PPP problems.

TABLE 7–1 Task Map for Troubleshooting PPP

Task	Definition	For Instructions
Obtain diagnostic information about the PPP link	Use PPP diagnostic tools to obtain output for troubleshooting.	"How to Obtain Diagnostic Information From `pppd`" on page 95
Obtain debugging information for the PPP link	Use the `pppd debug` command to generate output for troubleshooting.	"How to Turn on PPP Debugging" on page 96
Troubleshoot general problems with the network layer	Identify and fix PPP problems that are network-related by using a series of checks.	"How to Diagnose Network Problems" on page 97

TABLE 7–1 Task Map for Troubleshooting PPP *(Continued)*

Task	Definition	For Instructions
Troubleshoot general communications problems	Identify and fix communications problems that affect the PPP link.	"How to Diagnose and Fix Communications Problems" on page 100
Troubleshoot configuration problems	Identify and fix problems in the PPP configuration files.	"How to Diagnose Problems With the PPP Configuration" on page 101
Troubleshoot modem-related problems	Identify and fix modem problems.	"How to Diagnose Modem Problems" on page 102
Troubleshoot chat script-related problems	Identify and fix chat script problems on a dial-out machine.	"How to Obtain Debugging Information for Chat Scripts" on page 103
Troubleshoot serial-line speed problems	Identify and fix line–speed problems on a dial-in server.	"How to Diagnose and Fix Serial-Line Speed Problems" on page 105
Troubleshoot common problems for leased lines	Identify and fix performance problems on a leased line.	"Fixing Leased-Line Problems" on page 108
Troubleshoot problems related to authentication	Identify and fix problems related to the authentication databases.	"Diagnosing and Fixing Authentication Problems" on page 109
Troubleshoot problem areas for PPPoE	Use PPP diagnostic tools to obtain output for identifying and fixing PPPoE problems.	"How to Obtain Diagnostic Information for PPPoE" on page 106

Tools for Troubleshooting PPP

PPP links generally have three major areas of failure:

- Failure of the link to be established
- Poor performance of the link during regular usage
- Problems that can be traced to the networks on either side of the link

The easiest way to find out if PPP works is to run a command over the link. Run a command such as `ping` or `traceroute` to a host on the peer's network. Then observe the results. However, you should use PPP and UNIX debugging tools to monitor performance of an established link or to troubleshoot a problematic link.

This section explains how to obtain diagnostic information from `pppd` and its associated log files. The remaining sections in this chapter describe common problems with PPP that you can discover and fix with the aid of the PPP troubleshooting tools.

How to Obtain Diagnostic Information From pppd

The next procedure shows how to view the current operation of a link on the local machine.

1. **Become an administrator on the local machine.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Run pppd with the serial device configured for PPP as the argument:**

   ```
   # pppd cua/b debug updetach
   ```

   The next examples show the resulting displays for a dial-up link and a leased-line link when pppd runs in the foreground. If you run pppd debug in the background, the output that is produced is sent to the /etc/ppp/connect-errors file.

Example 7–1 Output From a Properly Operating Dial-up Link

```
# pppd /dev/cua/b debug updetach
have route to 0.0.0.0/0.0.0.0 via 172.21.0.4
serial speed set to 230400 bps
Using interface sppp0
Connect: sppp0 <--> /dev/cua/b
sent [LCP ConfReq id=0x7b <asyncmap 0x0> <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP Ident id=0x79 magic=0x0 "ppp-2.4.0b1 (Sun Microsystems, Inc., Oct  6
    2004 09:36:22)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Oct  6 2004 09:36:22)
    rcvd [LCP ConfRej id=0x7b <asyncmap 0x0>]
sent [LCP Ident id=0x7c magic=0x0 "ppp-2.4.0b1 (Sun Microsystems, Inc., Sep 15
    2004 09:38:33)"
sent [LCP ConfReq id=0x7d <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x7d <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x78 <magic 0xdd4ad820> <pcomp> <accomp>]
sent [LCP ConfAck id=0x78 <magic 0xdd4ad820> <pcomp> <accomp>]
sent [LCP Ident id=0x7e magic=0x73e981c8 "ppp-2.4.0b1 (Sun Microsystems, Inc.,
    Sep 15 2004 09:38:33)"]
sent [IPCP ConfReq id=0x3d <addr 0.0.0.0> <compress VJ 0f 01>]
rcvd [LCP Ident id=0x7a magic=0xdd4ad820 "ppp-2.4.0b1 (Sun Microsystems, Inc.,
    Oct  6 2004 09:36:22)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Oct  6 2004 09:36:22)
rcvd [IPCP ConfReq id=0x92 <addr 10.0.0.1> <compress VJ 0f 01>
sent [IPCP ConfAck id=0x92 <addr 10.0.0.1> <compress VJ 0f 01>
rcvd [IPCP ConfNak id=0x3d <addr 10.0.0.2>]]
sent [IPCP ConfReq id=0x3e <addr 10.0.0.2> <compress VJ 0f 01>]
rcvd [IPCP ConfAck id=0x3e <addr 10.0.0.2> <compress VJ 0f 01>]
local  IP address 10.0.0.2
remote IP address 10.0.0.1
```

Example 7–2 Output From a Properly Operating Leased-Line Link

```
# pppd /dev/se_hdlc1 default-asyncmap debug updetach
pppd 2.4.0b1 (Sun Microsystems, Inc., Oct 24 2004 07:13:18) started by root, uid 0
synchronous speed appears to be 0 bps
```

```
init option: '/etc/ppp/peers/syncinit.sh' started (pid 105122)
Serial port initialized.
synchronous speed appears to be 64000 bps
Using interface sppp0
Connect: sppp0 <--> /dev/se_hdlc1
sent [LCP ConfReq id=0xe9 <magic 0x474283c6><pcomp> <accomp>]
rcvd [LCP ConfAck id=0xe9 <magic 0x474283c6><pcomp> <accomp>]
rcvd [LCP ConfReq id=0x22 <magic 0x8e3a53ff><pcomp> <accomp>]
sent [LCP ConfReq id=0x22 <magic 0x8e3a53ff><pcomp> <accomp>]
sent [LCP Ident id=0xea magic=0x474283c6 "ppp-2.4.0b1 (Sun Microsystems, Inc., Oct
    22 2004 14:31:44)"]
sent [IPCP ConfReq id=0xf7 <addr 0.0.0.0> <compress VJ Of o1>]]
sent [CCP ConfReq id=0x3f <deflate 15> <deflate(old#) 15> <bsd v1 15>]
rcvd [LCP Ident id=0x23 magic=0x8e3a53ff "ppp-2.4.0b1 (Sun Microsystems, Inc., Oct
    22 2004 14:31:44)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Oct 22 2004 14:31:44)
rcvd [IPCP ConfReq id=0x25 <addr 10.0.0.1> <compress VJ Of 01>]
sent [IPCP ConfAck id=0x25 <addr 10.0.0.1> <compress VJ Of 01>]
rcvd [CCP ConfReq id=0x3 <deflate 15> <deflate(old#) 15 <bsd v1 15>]
sent [CCP ConfAck id=0x3 <deflate 15> <deflate(old#) 15 <bsd v1 15>]
rcvd [IPCP ConfNak id=0xf8 <addr 10.0.0.2>]
rcvd [IPCP ConfReq id=0xf7 <addr 10.0.0.2> <compress VJ Of 01>]
rcvd [CCP ConfAck id=0x3f <deflate 15> <deflate(old#) 15 <bsd v1 15>]
Deflate (15) compression enabled
rcvd [IPCP ConfAck id=0xf8 <addr 10.0.0.2> <compress VJ Of 01>]
local  IP address 10.0.0.2
remote IP address 10.0.0.1
```

How to Turn on PPP Debugging

The next task shows how to use the `pppd` command to obtain debugging information.

Note – You only need to perform step 1 through step 3 once for each host. Thereafter, you can proceed to step 4 to turn on debugging for the host.

1. **Become an administrator.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Create a log file to hold output from `pppd`.**

   ```
   # touch /var/log/pppdebug
   ```

3. **Add the following `syslog` facilities for `pppd` in `/etc/syslog.conf`.**

   ```
   daemon.debug;local2.debug       /var/log/pppdebug
   ```

4. **Restart `syslogd`.**

   ```
   # pkill -HUP -x syslogd
   ```

5 **Turn on debugging for calls to a particular peer by using the following syntax of pppd.**

```
# pppd debug call peer-name
```

peer-name must be the name of a file in the `/etc/ppp/peers` directory.

6 **View the contents of the log file.**

```
# tail -f /var/log/pppdebug
```

For an example of a log file, see Step 3.

Solving PPP-Related and PPPoE-Related Problems

Refer to the following sections for information about how to resolve PPP-related and PPPoE-related problems.

▼ How to Diagnose Network Problems

If the PPP link becomes active but few hosts on the remote network are reachable, a network problem could be indicated. The following procedure shows you how to isolate and fix network problems that affect a PPP link.

1 **Become an administrator on the local machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Shut down the problematic link.**

3 **Disable any optional protocols in the configuration files by adding the following options to your PPP configuration:**

```
noccp novj nopcomp noaccomp default-asyncmap
```

These options provide the simplest uncompressed PPP that is available. Try to invoke these options as arguments to `pppd` on the command line. If you can reach the previously unreachable hosts, add the options in either of the following places.

- `/etc/ppp/peers/`*peer-name*, after the `call` option
- `/etc/ppp/options`, ensuring that the options apply globally

4 Call the remote peer. Then enable debugging features.

```
% pppd debug call peer-name
```

5 Obtain verbose logs from the chat program by using the `-v` option of `chat`.

For example, use the following format in any PPP configuration file:

```
connect 'chat -v -f /etc/ppp/chatfile'
```

/etc/ppp/chatfile represents the name of your chat file.

6 Try to re-create the problem by using Telnet or other applications to reach the remote hosts.

Observe the debugging logs. If you still cannot reach remote hosts, the PPP problem might be network-related.

7 Verify that the IP addresses of the remote hosts are registered Internet addresses.

Some organizations assign internal IP addresses that are known within the local network but cannot be routed to the Internet. If the remote hosts are within your company, you must set up a name-to-address translation (NAT) server or proxy server to reach the Internet. If the remote hosts are not within your company, you should report the problem to the remote organization.

8 Examine the routing tables.

a. Check the routing tables on both the local machine and the peer.

b. Check the routing tables for any routers that are in the path from the peer to the remote system. Also check the routing tables for any routers on the path back to the peer.

Ensure that the intermediate routers have not been misconfigured. Often the problem can be found in the path back to the peer.

9 (Optional) If the machine is a router, check the optional features.

```
# ndd -set /dev/ip ip_forwarding 1
```

For more information about `ndd`, refer to the `ndd`(1M) man page.

In the Solaris 10 release, you can use `routeadm`(1M), instead of `ndd`(1M).

```
# routeadm -e ipv4-forwarding -u
```

Note – The `ndd` command is not persistent. The values set with this command are lost when the system is rebooted. The `routeadm` command is persistent. The values set with this command are maintained after the system is rebooted.

10 Check the statistics that are obtained from `netstat -s` and similar tools.

For complete details about `netstat`, refer to the `netstat(1M)` man page.

a. Run statistics on the local machine.

b. Call the peer.

c. Observe the new statistics that are generated by `netstat -s`.

For more information, refer to "Common Network Problems That Affect PPP" on page 99.

11 Check the DNS configuration.

A faulty name service configuration causes applications to fail because IP addresses cannot be resolved.

Common Network Problems That Affect PPP

You can use the messages that are generated by `netstat -s` to fix the network problems that are shown in the following table. For related procedural information, refer to "How to Diagnose Network Problems" on page 97.

TABLE 7–2 Common Network Problems That Affect PPP

Message	Problem	Solution
`IP packets not forwardable`	The local host is missing a route.	Add the missing route to the local host's routing tables.
`ICMP input destination unreachable`	The local host is missing a route.	Add the missing route to the local host's routing tables.
`ICMP time exceeded`	Two routers are forwarding the same destination address to each other, causing the packet to bounce back and forth until the time-to-live (TTL) value is exceeded.	Use `traceroute` to find the source of the routing loop, and then contact the administrator of the router in error. For information about `traceroute`, refer to the `traceroute(1M)` man page.
`IP packets not forwardable`	The local host is missing a route.	Add the missing route to the local host's routing table.
`ICMP input destination unreachable`	The local host is missing a route.	Add the missing route to the local host's routing tables.

▼ How to Diagnose and Fix Communications Problems

Communications problems occur when the two peers cannot successfully establish a link. Sometimes these problems are actually negotiation problems that are caused by incorrectly configured chat scripts. The following procedure shows you how to clear communication problems. For clearing negotiation problems that are caused by a faulty chat script, see Table 7–5.

1. **Become an administrator on the local machine.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Call the peer.**

3. **Call the remote peer. Then enable debugging features.**

   ```
   % pppd debug call peer-name
   ```

   You might need to obtain debugging information from the peer in order to fix certain communications problems.

4. **Check the resulting logs for communication problems. For more information, refer to "General Communications Problems That Affect PPP" on page 100.**

General Communications Problems That Affect PPP

The following table describes symptoms that are related to log output from the procedure, "How to Diagnose and Fix Communications Problems" on page 100.

TABLE 7–3 General Communications Problems That Affect PPP

Symptom	Problem	Solution
`too many Configure-Requests`	One peer cannot hear the other peer.	Check for the following problems: ■ The machine or modem might have faulty cabling. ■ The modem configuration might have incorrect bit settings. Or, the configuration might have broken flow control. ■ The chat script might have failed. In this situation, see Table 7–5.
The `pppd debug` output shows that LCP starts, but higher-level protocols fail or show CRC errors.	The asynchronous control character map (ACCM) is incorrectly set.	Use the `default-async` option to set the ACCM to the standard default of FFFFFFFF. First, try to use `default-async` as an option to `pppd` on the command line. If the problem clears, then add `default-async` to `/etc/ppp/options` or to `/etc/ppp/peers/`*peer-name* after the call option.

TABLE 7–3 General Communications Problems That Affect PPP *(Continued)*

Symptom	Problem	Solution
The `pppd debug` output shows that IPCP starts but terminates immediately.	IP addresses might be incorrectly configured.	1. Check the chat script to verify whether the script has incorrect IP addresses. 2. If the chat script is correct, request debug logs for the peer, and check IP addresses in the peer logs.
The link exhibits very poor performance.	The modem might be incorrectly configured, with flow-control configuration errors, modem setup errors, and incorrectly configured DTE rates.	Check the modem configuration. Adjust the configuration if necessary.

How to Diagnose Problems With the PPP Configuration

Some PPP problems can be traced to problems in the PPP configuration files. The following procedure shows you how to isolate and fix general configuration problems.

1 **Become an administrator on the local machine.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Call the remote peer. Then enable debugging features.**

```
% pppd debug call peer-name
```

3 **Check the resulting log for the configuration problems. For more information, refer to "Common PPP Configuration Problems" on page 101.**

Common PPP Configuration Problems

The following table describes symptoms that are related to log output from the procedure, "How to Diagnose Problems With the PPP Configuration" on page 101.

TABLE 7–4 Common PPP Configuration Problems

Symptom	Problem	Solution
`pppd debug` output contains the error message, `Could not determine remote IP address`.	The `/etc/ppp/peers/`*peer-name* file does not have an IP address for the peer. The peer does not provide an IP address during link negotiation.	Supply an IP address for the peer on the pppd command line or in `/etc/ppp/peers/`*peer-name* by using the following format: :10.0.0.10
`pppd debug` output shows that CCP data compression has failed. The output also indictes that the link is dropped.	The peers' PPP compression configurations might be in conflict.	Disable CCP compression by adding the `noccp` option to `/etc/ppp/options` on one of the peers.

How to Diagnose Modem Problems

Modems can be major problem areas for a dial-up link. The most common indicator of problems with the modem configuration is no response from the peer. However, you might have difficulties when determining if a link problem is indeed the result of modem configuration problems.

Modem manufacturers' documentation and web sites contain solutions for problems with their particular equipment. The following procedure helps determine whether a faulty modem configuration causes link problems.

1. **Call the peer with debugging turned on, as explained in "How to Turn on PPP Debugging" on page 96.**

2. **Display the resulting `/var/log/pppdebug` log to check for faulty modem configuration.**

3. **Use `ping` to send packets of various sizes over the link.**

   For complete details about `ping`, refer to the `ping(1M)` man page.

   If small packets are received but larger packets are dropped, modem problems are indicated.

4. **Check for errors on interface `sppp0`:**

```
% netstat -ni
Name  Mtu  Net/Dest   Address      Ipkts    Ierrs Opkts    Oerrs Collis Queue
lo0   8232 127.0.0.0  127.0.0.1    826808   0     826808   0     0      0
hme0  1500 172.21.0.0 172.21.3.228 13800032 0     1648464  0     0      0
sppp0 1500 10.0.0.2   10.0.0.1     210      0     128      0     0      0
```

   If interface errors increase over time, the modem configuration might have problems.

Troubleshooting When you display the resulting `/var/log/pppdebug` log, the following symptoms in the output can indicate a faulty modem configuration. The local machine can hear the peer, but the peer cannot hear the local machine.

- No "`recvd`" messages have come from the peer.
- The output contains LCP messages from the peer, but the link fails with `too many LCP Configure Requests` messages that are sent by the local machine.
- The link terminates with a SIGHUP signal.

How to Obtain Debugging Information for Chat Scripts

Use the following procedure for obtaining debugging information from `chat` and suggestions for clearing common problems. For more information, refer to "Common Chat Script Problems" on page 103.

1. **Become an administrator on the dial-out machine.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Edit the `/etc/ppp/peers/`*peer-name* file for the peer to be called.**

3. **Add `-v` as an argument to the `chat` command that is specified in `connect` option.**

   ```
   connect "/usr/bin/chat -v -f /etc/ppp/chat-script-name"
   ```

4. **View chat script errors in the file `/etc/ppp/connect-errors`.**

   The following is the main error that occurs with `chat`.

   ```
   Oct 31 08:57:13 deino chat[107294]: [ID 702911 local2.info] expect (CONNECT)
   Oct 31 08:57:58 deino chat[107294]: [ID 702911 local2.info] alarm
   Oct 31 08:57:58 deino chat[107294]: [ID 702911 local2.info] Failed
   ```

   The example shows timeout while waiting for a `(CONNECT)` string. When `chat` fails, you get the following message from `pppd`:

   ```
   Connect script failed
   ```

Common Chat Script Problems

Chat scripts are trouble-prone areas for dial-up links. The following table lists common chat script errors and gives suggestions for fixing the errors. For procedural information, refer to "How to Obtain Debugging Information for Chat Scripts" on page 103.

TABLE 7–5 Common Chat Script Problems

Symptom	Problem	Solution
`pppd debug` output contains `Connect script failed`	Your chat script supplies a user name and password. `ogin:` *user-name* `ssword:` *password* However, the peer that you intended to connect to does not prompt for this information.	1. Delete the login and password from the chat script. 2. Try to call the peer again. 3. If you still get the message, call the ISP. Ask the ISP for the correct login sequence.

The `/usr/bin/chat -v` log contains `"expect (login:)" alarm read timed out`	Your chat script supplies a user name and password.
	`ogin: pppuser`
	`ssword: \q\U`
	However, the peer that you intend to connect to does not prompt for this information.
		1. Delete the login and password from the chat script.
		2. Try to call the peer again.
		3. If you still get the message, call the ISP. Ask the ISP for the correct login sequence.

`pppd debug` output contains `possibly looped-back`	The local machine or its peer is hanging at the command line and not running PPP. An incorrectly configured login name and password are in the chat script.	1. Delete the login and password from the chat script. 2. Try to call the peer again. 3. If you still get the message, call the ISP. Ask for the correct login sequence.
`pppd debug` output shows that LCP activates, but the link terminates soon afterward.	The password in the chat script might be incorrect.	1. Ensure that you have the correct password for the local machine. 2. Check the password in the chat script. Fix the password if incorrect. 3. Try to call the peer again. 4. If you still get the message, call the ISP. Ask the ISP for the correct login sequence.

Text from the peer begins with a tilde (~).	Your chat script supplies a user name and password.
	`ogin: pppuser`
	`ssword: \q\U`
	However, the peer that you intend to connect to does not prompt for this information.
		1. Delete the login and password from the chat script.
		2. Try to call the peer again.
		3. If you still get the message, call the ISP. Request the correct login sequence.

TABLE 7–5 Common Chat Script Problems *(Continued)*

Symptom	Problem	Solution
The modem hangs.	Your chat script contains the following line to force the local machine to wait for the `CONNECT` message from the peer: `CONNECT ”`	Use the following line when you want the chat script to wait for CONNECT from the peer: `CONNECT \c` End the chat script with ~ `\c`.
`pppd debug output contains LCP: timeout sending Config-Requests`	Your chat script contains the following line to force the local machine to wait for the `CONNECT` message from the peer: `CONNECT ”`	Use the following line when you want the chat script to wait for CONNECT from the peer: `CONNECT \c` End the chat script with ~ `\c`.
`pppd debug output contains Serial link is not 8-bit clean`	Your chat script contains the following line to force the local machine to wait for the `CONNECT` message from the peer: `CONNECT ”`	Use the following line when you want the chat script to wait for CONNECT from the peer: `CONNECT \c` End the chat script with ~ `\c`.
`pppd debug output contains Loopback detected`	Your chat script contains the following line to force the local machine to wait for the `CONNECT` message from the peer: `CONNECT ”`	Use the following line when you want the chat script to wait for CONNECT from the peer: `CONNECT \c` End the chat script with ~ `\c`.
`pppd debug output contains SIGHUP`	Your chat script contains the following line to force the local machine to wait for the `CONNECT` message from the peer: `CONNECT ”`	Use the following line when you want the chat script to wait for CONNECT from the peer: `CONNECT \c` End the chat script with ~ `\c`.

▼ How to Diagnose and Fix Serial-Line Speed Problems

Dial-in servers can experience problems because of conflicting speed settings. The following procedure helps you to isolate the cause of the link problem to conflicting serial-line speeds.

The following behaviors cause speed problems:

- You invoked PPP through a program such as `/bin/login` and specified the speed of the line.
- You started PPP from `mgetty` and accidentally supplied the bit rate.

pppd changes the speed that was originally set for the line to the speed that was set by `/bin/login` or `mgetty`. As a result, the line fails.

1 Log in to the dial-in server. Call the peer with debugging enabled.

If you need instructions, see “How to Turn on PPP Debugging” on page 96.

2. **Display the resulting `/var/log/pppdebug` log.**

   Check the output for the following message:

   ```
   LCP too many configure requests
   ```

   This message indicates that the speeds of serial lines that were configured for PPP might potentially be in conflict.

3. **Check if PPP is invoked through a program such as `/bin/login` and the line speed that was set.**

   In such a situation, `pppd` changes the originally configured line speed to the speed that is specified in `/bin/login`.

4. **Check if a user started PPP from the `mgetty` command and accidentally specified a bit rate.**

   This action also causes serial-line speeds to conflict.

5. **Fix the conflicting serial-line speed problem as follows:**

   a. **Lock the DTE rate on the modem.**

   b. **Do not use autobaud.**

   c. **Do not change the line speed after configuration.**

How to Obtain Diagnostic Information for PPPoE

You can use PPP and standard UNIX utilities to identify problems with PPPoE. When you suspect that PPPoE is the cause of problems on a link, use the following diagnostic tools to obtain troubleshooting information.

1. **Become superuser on the machine that runs the PPPoE tunnel, either PPPoE client or PPPoE access server.**

2. **Turn on debugging, as explained in the procedure "How to Turn on PPP Debugging" on page 96.**

3. **View the contents of the log file `/var/log/pppdebug`.**

   The following example shows part of a log file that was generated for a link with a PPPoE tunnel.

```
Sep  6 16:28:45 enyo pppd[100563]: [ID 702911 daemon.info] Plugin
  pppoe.so loaded.
Sep  6 16:28:45 enyo pppd[100563]: [ID 860527 daemon.notice] pppd
  2.4.0b1 (Sun Microsystems, Inc.,
Sep  5 2001 10:42:05) started by troot, uid 0
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] connect option:
   '/usr/lib/inet/pppoec
-v hme0' started (pid 100564)
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.info] Serial connection established.
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.info] Using interface sppp0
```

```
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.notice] Connect: sppp0
   <--> /dev/sppptun
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] /etc/ppp/pap-secrets
  is apparently empty
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] /etc/ppp/chap-secrets
  is apparently empty
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] sent
  [LCP ConfReq id=0xef <mru 1492>
asyncmap 0x0 <magic 0x77d3e953><pcomp><acomp>
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] rcvd
  [LCP ConfReq id=0x2a <mru 1402>
asyncmap 0x0 <magic 0x9985f048><pcomp><acomp
```

If the debugging output does not help you isolate the problem, continue with this procedure.

4 Get diagnostic messages from PPPoE.

```
# pppd connect "/usr/lib/inet/pppoec -v interface-name"
```

`pppoec` sends diagnostic information to the `stderr`. If you run `pppd` in the foreground, the output appears on the screen. If `pppd` runs in the background, the output is sent to `/etc/ppp/connect-errors`.

The next example shows the messages that are generated as the PPPoE tunnel is negotiated.

```
Connect option: '/usr/lib/inet/pppoec -v hme0' started (pid 100564)
/usr/lib/inet/pppoec: PPPoE Event Open (1) in state Dead (0): action SendPADI (2)
/usr/lib/inet/pppoec: Sending PADI to ff:ff:ff:ff:ff:ff: 18 bytes
/usr/lib/inet/pppoec: PPPoE State change Dead (0) -> InitSent (1)
/usr/lib/inet/pppoec: Received Active Discovery Offer from 8:0:20:cd:c1:2/hme0:pppoed
/usr/lib/inet/pppoec: PPPoE Event rPADO+ (5) in state InitSent (1): action SendPADR+ (5)
/usr/lib/inet/pppoec: Sending PADR to 8:0:20:cd:c1:2: 22 bytes
/usr/lib/inet/pppoec: PPPoE State change InitSent (1) -> ReqSent (3)
/usr/lib/inet/pppoec: Received Active Discovery Session-confirmation from
   8:0:20:cd:c1:2/hme0:pppoed
/usr/lib/inet/pppoec: PPPoE Event rPADS (7) in state ReqSent (3): action Open (7)
/usr/lib/inet/pppoec: Connection open; session 0002 on hme0:pppoe
/usr/lib/inet/pppoec: PPPoE State change ReqSent (3) -> Convers (4)
/usr/lib/inet/pppoec: connected
```

If the diagnostic messages do not help you isolate the problem, continue with this procedure.

5 Run `snoop`. Then save the trace to a file.

For information about `snoop`, refer to the `snoop(1M)` man page.

```
# snoop -o pppoe-trace-file
```

6 View the `snoop` trace file.

```
# snoop -i pppoe-trace-file -v pppoe

ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 1 arrived at 6:35:2.77
ETHER: Packet size = 32 bytes
ETHER: Destination = ff:ff:ff:ff:ff:ff, (broadcast)
ETHER: Source      = 8:0:20:78:f3:7c, Sun
```

```
ETHER: Ethertype = 8863 (PPPoE Discovery)
ETHER:
PPPoE: ----- PPP Over Ethernet -----
PPPoE:
PPPoE: Version = 1
PPPoE: Type = 1
PPPoE: Code = 9 (Active Discovery Initiation)
PPPoE: Session Id = 0
PPPoE: Length = 12 bytes
PPPoE:
PPPoE: ----- Service-Name -----
PPPoE: Tag Type = 257
PPPoE: Tag Length = 0 bytes
PPPoE:
PPPoE: ----- Host-Uniq -----
PPPoE: Tag Type = 259
PPPoE: Tag Length = 4 bytes
PPPoE: Data = Ox00000002
PPPoE:
.
.
.
ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 5 arrived at 6:35:2.87
ETHER: Packet size = 60 bytes
ETHER: Destination = 8:0:20:78:f3:7c, Sun)
ETHER: Source      = 0:2:fd:39:7f:7,
ETHER: Ethertype = 8864 (PPPoE Session)
ETHER:
PPPoE: ----- PPP Over Ethernet -----
PPPoE:
PPPoE: Version = 1
PPPoE: Type = 1
PPPoE: Code = 0 (PPPoE Session)
PPPoE: Session Id = 24383
PPPoE: Length = 20 bytes
PPPoE:
PPP: ----- Point-to-Point Protocol -----
PPP:
PPP-LCP: ----- Link Control Protocol -----
PPP-LCP:
PPP-LCP: Code = 1 (Configure Request)
PPP-LCP: Identifier = 80
PPP-LCP: Length = 18
```

Fixing Leased-Line Problems

The most common problem with leased lines is poor performance. In most situations, you need to work with the telephone company to fix the problem.

TABLE 7–6 Common Leased-Line Problems

Symptom	Problem	Solution
The link does not start.	CSU biopolar violations (CSU BPVs) can be the cause. One end of the link is set up for AMI lines. The other end is set up for ESF bit–8 zero substitute (B8Zs).	If you are in the United States or Canada, you can directly fix this problem from the menu of the CSU/DSU. Check the CSU/DSU manufacturer's documentation for details. In other locales, the provider might be responsible for fixing CSU BPVs.
The link has poor performance.	The `pppd debug` output shows CRC errors when sustained traffic is on the link. Your line might have a clocking problem, caused by misconfigurations between the telephone company and your network.	Contact the telephone company to ensure that "loop clocking" is in use. On some unstructured leased lines, you might have to supply clocking. North American users should use loop clocking.

Diagnosing and Fixing Authentication Problems

The following table describes solutions for general authentication problems.

TABLE 7–7 General Authentication Problems

Symptom	Problem	Solution
`pppd debug` output shows the message `Peer is not authorized to use remote address` *address*.	You are using PAP authentication, and the IP address for the remote peer is not in the `/etc/ppp/pap-secrets` file.	Add an asterisk (`*`) after the entry for the peer in the `/etc/ppp/pap-secrets` file.
`pppd debug` output shows that LCP starts but terminates shortly afterward.	The password might be incorrect in the database for the particular security protocol.	Check the password for the peer in the `/etc/ppp/pap-secrets` or `/etc/ppp/chap-secrets` file.

◆ ◆ ◆ CHAPTER 8

Solaris PPP 4.0 (Reference)

This chapter provides detailed conceptual information about Solaris PPP 4.0. Topics include the following:

Using PPP Options in Files and on the Command Line

Solaris PPP 4.0 contains a large set of options, which you use to define your PPP configuration. You use these options in the PPP configuration files, or on the command line, or by using a combination of files and command-line options. This section contains detailed information about the use of PPP options in configuration files and as arguments to PPP commands.

Where to Define PPP Options

Solaris PPP 4.0 configuration is very flexible. You can define PPP options in the following places:

- PPP configuration files
- PPP commands that are issued on the command line
- A combination of both places

The next table lists the PPP configuration files and commands.

TABLE 8–1 Summary of PPP Configuration Files and Commands

File or Command	Description	For Information
`/etc/ppp/options`	A file that contains characteristics that apply by default to all PPP links on the system, for example, whether the machine requires peers to authenticate themselves. If this file is absent, nonroot users are prohibited from using PPP.	"`/etc/ppp/options` Configuration File" on page 115
`/etc/ppp/options.`*ttyname*	A file that describes the characteristics of all communications over the serial port *ttyname*.	"`/etc/ppp/options.`*ttyname* Configuration File" on page 116
`/etc/ppp/peers`	Directory that usually contains information about peers with which a dial-out machine connects. Files in this directory are used with the `call` option of the `pppd` command.	"Specifying Information for Communicating With the Dial-in Server" on page 119
`/etc/ppp/peers/`*peer-name*	A file that contains characteristics of the remote peer *peer-name*. Typical characteristics include the remote peer's phone number and chat script for negotiating the link with the peer.	"`/etc/ppp/peers/`*peer-name* File" on page 120
`/etc/ppp/pap-secrets`	A file that contains the necessary security credentials for Password Authentication Protocol (PAP) authentication.	"`/etc/ppp/pap-secrets` File" on page 132
`/etc/ppp/chap-secrets`	A file that contains the necessary security credentials for Challenge-Handshake Authentication Protocol (CHAP) authentication.	"`/etc/ppp/chap-secrets` File" on page 135
`~/.ppprc`	File in the home directory of a PPP user, most often used with dial-in servers. This file contains specific information about each user's configuration.	"Configuring `$HOME/.ppprc` on a Dial-in Server" on page 119
`pppd` *options*	Command and options for initiating a PPP link and describing its characteristics.	"How PPP Options Are Processed" on page 112

Refer to the pppd(1M) man page for details on the PPP files. `pppd` (1M) also includes comprehensive descriptions of all options that are available to the `pppd` command. Sample templates for all the PPP configuration files are available in `/etc/ppp`.

How PPP Options Are Processed

1. The `pppd` daemon parses the following:

All Solaris PPP 4.0operations are handled by the `pppd` daemon, which starts when a user runs the `pppd` command. When a user calls a remote peer, the following occurs:

- `/etc/ppp/options`
- `$HOME/.ppprc`
- Any files that are opened by the `file` or `call` option in `/etc/ppp/options` and `$HOME/.ppprc`

2. `pppd` scans the command line to determine the device in use. The daemon does not yet interpret any options that are encountered.
3. `pppd` tries to discover the serial device to use by using these criteria:
   - If a serial device is specified on the command line, or a previously processed configuration file, `pppd` uses the name of that device.
   - If no serial device is named, then `pppd` searches for the `notty`, `pty`, or `socket` option on the command line. If one of these options is specified, `pppd` assumes that no device name exists.
   - Otherwise, if `pppd` discovers that standard input is attached to a tty, then the name of the tty is used.
   - If `pppd` still cannot find a serial device, `pppd` terminates the connection and issues an error.
4. `pppd` then checks for the existence of the `/etc/ppp/options.`*ttyname* file. If the file is found, `pppd` parses the file.
5. `pppd` processes any options on the command line.
6. `pppd` negotiates the Link Control Protocol (LCP) to set up the link.
7. (Optional) If authentication is required, `pppd` reads `/etc/ppp/pap-secrets` or `/etc/ppp/chap-secrets` to authenticate the opposite peer.

The file `/etc/ppp/peers/`*peer-name* is read when the `pppd` daemon encounters the option `call` *peer-name* on the command line or in the other configuration files.

How PPP Configuration File Privileges Work

Solaris PPP 4.0 configuration includes the concept of *privileges*. Privileges determine the precedence of configuration options, particularly when the same option is invoked in more than one place. An option that is invoked from a privileged source takes precedence over the same option that is invoked from a nonprivileged source.

User Privileges

The only privileged user is superuser (`root`), with the UID of zero. All other users are not privileged.

File Privileges

The following configuration files are privileged regardless of their ownership:

- `/etc/ppp/options`
- `/etc/ppp/options.`*ttyname*
- `/etc/ppp/peers/`*peer-name*

The file `$HOME/.ppprc` is owned by the user. Options that are read from `$HOME/.ppprc` and from the command line are privileged only if the user who is invoking `pppd` is `root`.

Arguments that follow the `file` option are privileged.

Effects of Option Privileges

Some options require the invoking user or source to be privileged in order to work. Options that are invoked on the command line are assigned the privileges of the user who is running the `pppd` command. These options are not privileged unless the user who is invoking `pppd` is `root`.

Option	Status	Explanation
`domain`	Privileged	Requires privileges for use.
`linkname`	Privileged	Requires privileges for use.
`noauth`	Privileged	Requires privileges for use.
`nopam`	Privileged	Requires privileges for use.
`pam`	Privileged	Requires privileges for use.
`plugin`	Privileged	Requires privileges for use.
`privgroup`	Privileged	Requires privileges for use.
`allow-ip` *addresses*	Privileged	Requires privileges for use.
`name` *hostname*	Privileged	Requires privileges for use.
`plink`	Privileged	Requires privileges for use.
`noplink`	Privileged	Requires privileges for use.
`plumbed`	Privileged	Requires privileges for use.
`proxyarp`	Becomes privileged if `noproxyarp` has been specified	Cannot be overridden by an unprivileged user.
`defaultroute`	Privileged if `nodefaultroute` is set in a privileged file or by a privileged user	Cannot be overridden by an unprivileged user.
`disconnect`	Privileged if set in a privileged file or by a privileged user	Cannot be overridden by an unprivileged user.

Option	Status	Explanation
`bsdcomp`	Privileged if set in a privileged file or by a privileged user	The nonprivileged user cannot specify a code size that is larger than the privileged user has specified.
`deflate`	Privileged if set in a privileged file or by a privileged user	The nonprivileged user cannot specify a code size that is larger than the privileged user has specified.
`connect`	Privileged if set in a privileged file or by a privileged user	Cannot be overridden by an nonprivileged user.
`init`	Privileged if set in a privileged file or by a privileged user	Cannot be overridden by an nonprivileged user.
`pty`	Privileged if set in a privileged file or by a privileged user	Cannot be overridden by an nonprivileged user.
`welcome`	Privileged if set in a privileged file or by a privileged user	Cannot be overridden by an nonprivileged user.
ttyname	Privileged if set in a privileged file Not privileged if set in a nonprivileged file	Opened with root permissions regardless of who invokes `pppd`. Opened with the privileges of the user who invokes `pppd`.

`/etc/ppp/options` Configuration File

You use the `/etc/ppp/options` file to define global options for all PPP communications on the local machine. `/etc/ppp/options` is a privileged file. `/etc/ppp/options` should be owned by root, although `pppd` does not enforce this rule. Options that you define in `/etc/ppp/options` have precedence over definitions of the same options in all other files and the command line.

Typical options that you might use in `/etc/ppp/options` include the following:

- **`lock`** – Enables UUCP-style file locking
- **`noauth`** – Indicates that the machine does not authenticate callers

---

Note – The Solaris PPP 4.0 software does not include a default `/etc/ppp/options` file. `pppd` does not require the `/etc/ppp/options` file to work. If a machine does not have an `/etc/ppp/options` file, only `root` can run `pppd` on that machine.

You must create `/etc/ppp/options` by using a text editor, as shown in "How to Define Communications Over the Serial Line" on page 54. If a machine does not require global options, you can create an empty `/etc/ppp/options` file. Then, both `root` and regular users can run `pppd` on the local machine.

`/etc/ppp/options.tmpl` Template

The `/etc/ppp/options.tmpl` contains helpful comments about the `/etc/ppp/options` file plus three common options for the global `/etc/ppp/options` file.

```
lock
nodefaultroute
noproxyarp
```

Option	Definition
`lock`	Enables UUCP-style file locking
`nodefaultroute`	Specifies that no default route is defined
`noproxyarp`	Disallows `proxyarp`

To use `/etc/ppp/options.tmpl` as the global options file, rename `/etc/ppp/options.tmpl` to `/etc/ppp/options`. Then, modify the file contents as needed by your site.

Where to Find Examples of the `/etc/ppp/options` Files

To find examples of the `/etc/ppp/options` file, refer to the following:

- For a dial-out machine, see "How to Define Communications Over the Serial Line" on page 54.
- For a dial-in server, see "How to Define Communications Over the Serial Line (Dial-in Server)" on page 61.
- For PAP support on a dial-in server, see "How to Add PAP Support to the PPP Configuration Files (Dial-in Server)" on page 75.
- For PAP support on a dial-out machine, see "How to Add PAP Support to the PPP Configuration Files (Dial-out Machine)" on page 78.
- For CHAP support on a dial-in server, see "How to Add CHAP Support to the PPP Configuration Files (Dial-in Server)" on page 82.

`/etc/ppp/options.`*ttyname* Configuration File

You can configure the characteristics of communications on the serial line in the `/etc/ppp/options.`*ttyname* file. `/etc/ppp/options.`*ttyname* is a privileged file that is read by `pppd` after parsing any existing `/etc/ppp/options` and existing `$HOME/.pppprc` files. Otherwise, `pppd` reads `/etc/ppp/options.`*ttyname* after parsing `/etc/ppp/options`.

ttyname is used for both dial-up and leased-line links. *ttyname* represents a particular serial port on a machine, such as `cua/a` or `cua/b`, where a modem or ISDN TA might be attached.

When naming the `/etc/ppp/options.`*ttyname* file, replace the slash (/) in the device name with a dot (.). For example, the `options` file for device `cua/b` should be named `/etc/ppp/options.cua.b`.

Note – Solaris PPP 4.0 does not require an `/etc/ppp/options.` *ttyname* file to work correctly. Your server might have only one serial line for PPP. Furthermore, the server requires few options. In this instance, you can specify any required options in another configuration file or on the command line.

Using `/etc/ppp/options.` *ttyname* on a Dial-in Server

For a dial-up link, you might choose to create individual `/etc/ppp/options.` *ttyname* files for every serial port on a dial-in server with a modem attached. Typical options include the following:

- IP address required by the dial-in server

  Set this option if you require incoming callers on serial port *ttyname* to use a particular IP address. Your address space might have a limited number of IP addresses that are available for PPP in comparison to the number of potential callers. In this situation, consider assigning an IP address to each serial interface that is used for PPP on the dial-in server. This assignment implements dynamic addressing for PPP.

- `asyncmap` *map-value*

  The `asyncmap` option maps control characters that cannot be received over the serial line by the particular modem or ISDN TA. When the `xonxoff` option is used, `pppd` automatically sets an asyncmap of 0xa0000.

  map-value states, in hexadecimal format, the control characters that are problematic.

- `init "chat -U -f /etc/ppp/mychat"`

  The `init` option tells the modem to initialize communications over the serial line by using the information in the `chat -U` command. The modem uses the chat string in the file `/etc/ppp/mychat`.

- Security parameters that are listed in the `pppd`(1m) man page

Using `/etc/ppp/options.` *ttyname* on a Dial-out Machine

For a dial-out system, you can create an `/etc/ppp/options.` *ttyname* file for the serial port that is connected to the modem, or choose not to use `/etc/ppp/options.` *ttyname*.

Note – Solaris PPP 4.0 does not require an `/etc/ppp/options.`*ttyname* file to work correctly. A dial-out machine might have only one serial line for PPP. Furthermore, the dial-out machine might require few options. You can specify any required options in another configuration file or on the command line.

`options.ttya.tmpl` Template File

The `/etc/ppp/options.ttya.tmpl` file contains helpful comments about the `/etc/ppp/options.`*tty-name* file. The template contains three common options for the `/etc/ppp/options.`*tty-name* file.

```
38400
asyncmap 0xa0000
:192.168.1.1
```

Option	Definition
`38400`	Use this baud rate for port ttya.
`asyncmap 0xa0000`	Assign the asyncmap value of 0xa0000 so that the local machine can communicate with broken peers.
`:192.168.1.1`	Assign the IP address 192.168.1.1 to all peers that are calling in over the link.

To use `/etc/ppp/options.ttya.tmpl` at your site, rename `/etc/ppp/options.tmpl` to `/etc/ppp/options.`*ttya-name*. Replace *ttya-name* with the name of the serial port with the modem. Then modify the file contents as needed by your site.

Where to Find Examples of the `/etc/ppp/options.`*ttyname* Files

To find examples of the `/etc/ppp/options.`*ttyname* files, refer to the following:

- For a dial-out machine, see "How to Define Communications Over the Serial Line" on page 54.
- For a dial-in server, see "How to Define Communications Over the Serial Line (Dial-in Server)" on page 61.

Configuring User-Specific Options

This section contains detailed information about setting up users on the dial-in server.

Configuring `$HOME/.pppprc` on a Dial-in Server

The `$HOME/.pppprc` file is intended for users who are configuring preferred PPP options. As administrator, you can also configure `$HOME/.pppprc` for users.

The options in `$HOME/.pppprc` are privileged only when the user who is invoking the file is privileged.

When a caller uses the `pppd` command to initiate a call, the `.pppprc` file is the second file that is checked by the `pppd` daemon.

See "Setting Up Users of the Dial-in Server" on page 59 for instructions about setting up `$HOME/.pppprc` on the dial-in server.

Configuring `$HOME/.pppprc` on a Dial-out Machine

The `$HOME/.pppprc` file is not needed on the dial-out machine for Solaris PPP 4.0 to work correctly. Additionally, you do not need to have a `$HOME/.pppprc` on a dial-out machine, except for special circumstances. Create one or more `.pppprc` files if you do the following:

- Allow multiple users with differing communications needs to call remote peers from the same machine. In such an instance, create individual `.pppprc` files in the home directories of each user who must dial out.
- Need to specify options that control problems specific to your link, such as disabling Van Jacobson compression. See James Carlson's *PPP Design, Implementation, and Debugging* and the `pppd(1M)` man page for assistance in troubleshooting link problems.

Because the `.pppprc` file is most often used when configuring a dial-in server, refer to "How to Configure Users of the Dial-in Server" on page 60 for configuration instructions for `.pppprc`.

Specifying Information for Communicating With the Dial-in Server

To communicate with a dial-in server, you need to gather information about the server. Then edit a few files. Most significantly, you must configure the communications requirements of all dial-in servers that the dial-out machine needs to call. You can specify options about a dial-in server, such as an ISP phone number, in the `/etc/ppp/options.`*ttyname* file. However, the optimum place to configure peer information is in `/etc/ppp/peers/`*peer-name* files.

/etc/ppp/peers/*peer-name* File

Note – The /etc/ppp/peers/*peer-name* file is not needed on the dial-out machine for Solaris PPP 4.0 to work correctly.

Use the /etc/ppp/peers/*peer-name* file to provide information for communicating with a particular peer. /etc/ppp/peers/*peer-name* allows ordinary users to invoke preselected privileged options that users are not allowed to set.

For example, a nonprivileged user cannot override the noauth option if noauth is specified in the /etc/ppp/peers/*peer-name* file. Suppose the user wants to set up a link to peerB, which does not provide authentication credentials. As superuser, you can create a /etc/ppp/peers/peerB file that includes the noauth option. noauth indicates that the local machine does not authenticate calls from peerB.

The pppd daemon reads /etc/ppp/peers/*peer-name* when pppd encounters the following option:

call *peer-name*

You can create a /etc/ppp/peers/*peer-name* file for each target peer with which the dial-out machine needs to communicate. This practice is particularly convenient for permitting ordinary users to invoke special dial-out links without needing root privileges.

Typical options that you specify in /etc/ppp/peers/*peer-name* include the following:

- user *user-name*

  Supply *user-name* to the dial-in server, as the login name of the dial-out machine, when authenticating with PAP or CHAP.

- remotename *peer-name*

  Use *peer-name* as the name of the dial-in machine. remotename is used in conjunction with PAP or CHAP authentication when scanning the /etc/ppp/pap-secrets or /etc/ppp/chap-secrets files.

- connect "chat *chat_script*..."

  Open communication to the dial-in server by using the instructions in the chat script.

- noauth

  Do not authenticate the peer *peer-name* when initiating communications.

- noipdefault

  Set the initial IP address that is used in negotiating with the peer to 0.0.0.0. Use noipdefault when setting up a link to most ISPs to help facilitate IPCP negotiation between the peers.

- defaultroute

Install a default IPv4 route when IP is established on the link.

See the pppd(1M) man page for more options that might apply to a specific target peer.

/etc/ppp/peers/myisp.tmpl Template File

The /etc/ppp/peers/myisp.tmpl file contains helpful comments about the /etc/ppp/peers/*peer-name* file. The template concludes with common options that you might use for an /etc/ppp/peers/*peer-name* file:

```
connect "/usr/bin/chat -f /etc/ppp/myisp-chat"
user myname
remotename myisp
noauth
noipdefault
defaultroute
updetach
noccp
```

Option	Definition
`connect "/usr/bin/chat -f /etc/ppp/myisp-chat"`	Call the peer by using the chat script /etc/ppp/myisp-chat.
`user myname`	Use this account name for the local machine. myname is the name for this machine in the peer's /etc/ppp/pap-secrets file.
`remotename myisp`	Recognize myisp as the name of the peer in the local machine's /etc/ppp/pap-secrets file.
`noauth`	Do not require calling peers to provide authentication credentials.
`noipdefault`	Do not use a default IP address for the local machine.
`defaultroute`	Use the default route that is assigned to the local machine.
`updetach`	Log errors in the PPP log files, rather than on the standard output.
`noccp`	Do not use CCP compression.

To use /etc/ppp/peers/myisp.tmpl at your site, rename /etc/ppp/peers/myisp.tmpl to /etc/ppp/peers/.*peer-name*. Replace *peer-name* with the name of the peer to be called. Then modify the file contents as needed by your site.

Where to Find Examples of the /etc/ppp/peers/*peer-name* Files

To find examples of the /etc/ppp/peers/*peer-name* files, refer to the following:

- For a dial-out machine, see "How to Define the Connection With an Individual Peer" on page 56.
- For a local machine on a leased line, see "How to Configure a Machine on a Leased Line" on page 67.
- For support of PAP authentication on a dial-out machine, see "How to Add PAP Support to the PPP Configuration Files (Dial-out Machine)" on page 78.
- For support of CHAP authentication on a dial-out machine, see "How to Add CHAP Support to the PPP Configuration Files (Dial-out Machine)" on page 84.
- For support of PPPoE on a client system, see "Setting Up the PPPoE Client" on page 86.

Configuring Modem Speed for a Dial-up Link

A major issue in modem configuration is designating the speed at which the modem should operate. The following guidelines apply to modems that are used with Sun Microsystems computers:

- Older SPARC systems – Check the hardware documentation that accompanies the system. Many SPARCstation machines require modem speed not to exceed 38400 bps.
- UltraSPARC machines – Set the modem speed to 115200 bps, which is useful with modern modems and fast enough for a dial-up link. If you plan to use a dual-channel ISDN TA with compression, you need to increase the modem speed. The limit on an UltraSPARC is 460800 bps for an asynchronous link.

For a *dial-out machine*, set the modem speed in the PPP configuration files, such as /etc/ppp/peers/*peer-name*, or by specifying the speed as an option for pppd.

For a *dial-in server*, you need to set the speed by using the ttymon facility as described in "Configuring Devices on the Dial-in Server" on page 58.

Defining the Conversation on the Dial-up Link

The dial-out machine and its remote peer communicate across the PPP link by negotiating and exchanging various instructions. When configuring a dial-out machine, you need to determine what instructions are required by the local and remote modems. Then you create a file that is called a chat script that contains these instructions. This section discusses information about configuring modems and creating chat scripts.

Contents of the Chat Script

Each remote peer that the dial-out machine needs to connect to probably requires its own chat script.

Note – Chat scripts are typically used only on dial-up links. Leased-line links do not use chat scripts unless the link includes an asynchronous interface that requires startup configuration.

The contents of the chat script are determined by the requirements of your modem model or ISDN TA, and the remote peer. These contents appear as a set of expect-send strings. The dial-out machine and its remote peers exchange the strings as part of the communications initiation process.

An *expect* string contains characters that the dial-out host machine expects to receive from the remote peer to initiate conversation. A *send* string contains characters that the dial-out machine sends to the remote peer after receiving the expect string.

Information in the chat script usually includes the following:

- Modem commands, often referred to as *AT commands*, which enable the modem to transmit data over the telephone
- Phone number of the target peer

  This phone number might be the number that is required by your ISP, or a dial-in server at a corporate site, or an individual machine.
- Time-out value, if required
- Login sequence that is expected from the remote peer
- Login sequence that is sent by the dial-out machine

Chat Script Examples

This section contains chat scripts that you can use as a reference for creating your own chat scripts. The modem manufacturer's guide and information from your ISP and other target hosts contain chat requirements for the modem and your target peers. In addition, numerous PPP web sites have sample chat scripts.

Basic Modem Chat Script

The following is a basic chat script that you can use as a template for creating your own chat scripts.

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
```

```
TIMEOUT 10
"" AT&F1M0&M5S2=255
SAY     "Calling myserver\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
ogin: pppuser
ssword: \q\U
% pppd
```

The next table describes the contents of the chat script.

Script Contents	Explanation
`ABORT BUSY`	Abort transmission if the modem receives this message from the opposite peer.
`ABORT 'NO CARRIER'`	Abort transmission if the modem reports `ABORT 'NO CARRIER'` when dialing. The cause for this message is usually a dialing or modem negotiation failure.
`REPORT CONNECT`	Gather the `CONNECT` string from the modem. Print the string.
`TIMEOUT 10`	Set initial timeout to 10 seconds. The modem's response should be immediate.
`"" AT&F1M0&M5S2=255`	`M0` – Turn off the speaker during connect.
	`&M5` – Make the modem require error control.
	`S2=255` – Disable the TIES "+++" break sequence.
`SAY "Calling myserver\n"`	Display the message `Calling myserver` on the local machine.
`TIMEOUT 60`	Reset the timeout to 60 seconds to allow more time for link negotiation.
`OK "ATDT1-123-555-1212"`	Call the remote peer by using the phone number 123-555-1212.
`ogin: pppuser`	Log in to the peer by using UNIX-style login. Supply the user name `pppuser`.
`ssword: \q\U`	`\q` – Do not log if debugging with the `-v` option.
	`\U` – Insert in this location the contents of the string that follows `-U`, which is specified on the command line. Usually, the string contains the password.
`% pppd`	Wait for the % shell prompt, and run the `pppd` command.

`/etc/ppp/myisp-chat.tmpl` Chat Script Template

This release includes the `/etc/ppp/myisp-chat.tmpl`, which you can modify for use at your site. `/etc/ppp/myisp-chat.tmpl` is similar to the basic modem chat script except that the template does not include a login sequence.

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
TIMEOUT 10
""      "AT&F1"
OK      "AT&C1&D2"
```

```
SAY     "Calling myisp\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
CONNECT \c
```

Script Contents	Explanation
`ABORT BUSY`	Abort transmission if the modem receives this message from the opposite peer.
`ABORT 'NO CARRIER`	Abort transmission if the modem reports `ABORT 'NO CARRIER'` when dialing. The cause for this message is usually a dialing or modem negotiation failure.
`REPORT CONNECT`	Gather the `CONNECT` string from the modem. Print the string.
`TIMEOUT 10`	Set initial timeout to 10 seconds. The modem's response should be immediate.
`"" "AT&F1"`	Reset the modem to factory defaults.
`OK "AT&C1&D2"`	Reset the modem so that, for &C1, DCD from the modem follows carrier. If the remote side hangs up the phone for some reason, then the DCD drops. For &D2, DTR high-to-low transition causes the modem to go "on-hook" or hang up.
`SAY "Calling myisp\n"`	Display the message "Calling myisp" on the local machine.
`TIMEOUT 60`	Reset the timeout to 60 seconds to allow more time for link negotiation.
`OK "ATDT1-123-555-1212"`	Call the remote peer by using the phone number 123-555-1212.
`CONNECT \c`	Wait for the `CONNECT` message from the opposite peer's modem.

Modem Chat Script for Calling an ISP

Use the next chat script as a template for calling an ISP from a dial-out machine with a U.S. Robotics Courier modem.

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
TIMEOUT 10
"" AT&F1M0&M5S2=255
SAY     "Calling myisp\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
CONNECT \c
\r \d\c
SAY "Connected; running PPP\n"
```

The following table describes the contents of the chat script.

Script Contents	Explanation
`ABORT BUSY`	Abort transmission if the modem receives this message from the opposite peer.
`ABORT 'NO CARRIER'`	Abort transmission if the modem receives this message from the opposite peer.
`REPORT CONNECT`	Gather the `CONNECT` string from the modem. Print the string.
`TIMEOUT 10`	Set initial timeout to 10 seconds. The modem's response should be immediate.
`"" AT&F1M0M0M0M0&M5S2=255`	`M0` – Turn off the speaker during connect.
	`&M5` – Make the modem require error control.
	`S2=255` – Disable the TIES "+++" break sequence.
`SAY "Calling myisp\n"`	Display the message `Calling myisp` on the local machine.
`TIMEOUT 60`	Reset the timeout to 60 seconds to allow more time for link negotiation.
`OK "ATDT1-123-555-1212"`	Call the remote peer by using the phone number 123-555-1212.
`CONNECT \c`	Wait for the `CONNECT` message from the opposite peer's modem.
`\r \d\c`	Wait until the end of the `CONNECT` message.
SAY "Connected; running PPP\n"	Display the informative message `Connected; running PPP` on the local machine.

Basic Chat Script Enhanced for a UNIX-Style Login

The next chat script is a basic script that is enhanced for calling a remote Oracle Solaris peer or other UNIX-type peer. This chat script is used in "How to Create the Instructions for Calling a Peer" on page 55.

```
SAY "Calling the peer\n"
TIMEOUT 10
ABORT BUSY
ABORT 'NO CARRIER'
ABORT ERROR
REPORT CONNECT
"" AT&F1&M5S2=255
TIMEOUT 60
OK ATDT1-123-555-1234
CONNECT \c
SAY "Connected; logging in.\n"
TIMEOUT 5
ogin:--ogin: pppuser
TIMEOUT 20
ABORT 'ogin incorrect'
ssword: \qmypassword
"% " \c
SAY "Logged in.  Starting PPP on peer system.\n"
ABORT 'not found'
"" "exec pppd"
~ \c
```

The following table explains the parameters of the chat script.

Script Contents	Explanation
`TIMEOUT 10`	Set initial timeout to 10 seconds. The modem's response should be immediate.
`ABORT BUSY`	Abort transmission if the modem receives this message from the opposite peer.
`ABORT 'NO CARRIER'`	Abort transmission if the modem receives this message from the opposite peer.
`ABORT ERROR`	Abort transmission if the modem receives this message from the opposite peer.
`REPORT CONNECT`	Gather the `CONNECT` string from the modem. Print the string.
`"" AT&F1&M5S2=255`	`&M5` – Make the modem require error control. `S2=255` – Disable the TIES "+++" break sequence.
`TIMEOUT 60`	Reset the timeout to 60 seconds to allow more time for link negotiation.
`OK ATDT1-123-555-1234`	Call the remote peer by using the phone number 123-555-1212.
`CONNECT \c`	Wait for the `CONNECT` message from the opposite peer's modem.
`SAY "Connected; logging in.\n"`	Display the informative message `Connected; logging in` to give the user status.
`TIMEOUT 5`	Change the timeout to enable quick display of the login prompt.
`ogin:--ogin: pppuser`	Wait for the login prompt. If the prompt is not received, send a RETURN and wait. Then, send the user name `pppuser` to the peer. The sequence that follows is referred to by most ISPs as the PAP login. However, the PAP login is not related in any way to PAP authentication.
`TIMEOUT 20`	Change the timeout to 20 seconds to allow for slow password verification.
`ssword: \qmysecrethere`	Wait for the password prompt from the peer. When the prompt is received, send the password `\qmysecrethere`. The `\q` prevents the password from being written to the system log files.
`"% " \c`	Wait for a shell prompt from the peer. The chat script uses the C shell. Change this value if the user prefers to log in with a different shell.
`SAY "Logged in. Starting PPP on peer system.\n"`	Display the informative message `Logged in. Starting PPP on peer system` to give the user status.
`ABORT 'not found'`	Abort the transmission if the shell encounters errors.
`"" "exec pppd"`	Start `pppd` on the peer.

Script Contents	Explanation
~ \c	Wait for PPP to start on the peer.

Starting PPP right after the CONNECT \c is often called a *PAP login* by ISPs, though the PAP login is actually not part of PAP authentication.

The phrase ogin:--ogin: pppuser instructs the modem to send the user name pppuser in response to the login prompt from the dial-in server. pppuser is a special PPP user account name that was created for remote user1 on the dial-in server. For instructions about creating PPP user accounts on a dial-in server, refer to "How to Configure Users of the Dial-in Server" on page 60.

Chat Script for External ISDN TA

The following chat script is for calling from a dial-out machine with a ZyXEL omni.net. ISDN TA.

```
SAY "Calling the peer\n"
TIMEOUT 10
ABORT BUSY
ABORT 'NO CARRIER'
ABORT ERROR
REPORT CONNECT
"" AT&FB40S83.7=1&K44&J3X7S61.3=1S0=0S2=255
OK ATDI18882638234
CONNECT \c
\r \d\c
SAY "Connected; running PPP\n"
```

The following table explains the parameters of the chat script.

Script Contents	Explanation
SAY "Calling the peer"	Display this message on the screen of the dial-out machine.
TIMEOUT 10	Set the initial timeout to 10 seconds.
ABORT BUSY	Abort transmission if the modem receives this message from the opposite peer.
ABORT 'NO CARRIER'	Abort transmission if the modem receives this message from the opposite peer.
ABORT ERROR	Abort transmission if the modem receives this message from the opposite peer.
REPORT CONNECT	Gather the CONNECT string from the modem. Print the string.

Script Contents	Explanation
`"" AT&FB40S83.7=1&K44&J3X7S61.3=1S0=0S2=255`	The letters in this line have the following meaning: ■ `&F` – Use factory default ■ `B40` – Do asynchronous PPP conversion ■ `S83.7=1` – Use data over speech bearer ■ `&K44` – Enable CCP compression ■ `&J3` – Enable MP ■ `X7` – Report DCE side rates ■ `S61.3=1` – Use packet fragmentation ■ `S0=0` – No auto answer ■ `S2=255` – Disable TIES escape
`OK ATDI18882638234`	Make an ISDN call. For multilink, the second call is placed to the same telephone number, which is normally what is required by most ISPs. If the remote peer requires a different second phone number, append "+*nnnn*.". *nnnn* represents the second phone number.
`CONNECT \c`	Wait for the `CONNECT` message from the opposite peer's modem.
`\r \d\c`	Wait until the end of the `CONNECT` message.
`SAY "Connected; running PPP\n"`	Display this message on the screen of the dial-out machine.

Refer to the `chat(1M)` man page for descriptions of options and other detailed information about the chat script. For an explanation of expect-send strings, refer to "Chat-Script Field in `/etc/uucp/Systems` File" on page 175.

For More Chat Script Examples

A number of web sites offer sample chat scripts and assistance in creating the chat scripts. For example, see http://ppp.samba.org/ppp/index.html.

Invoking the Chat Script

You call chat scripts by using the `connect` option. You can use `connect "chat ..."` in any PPP configuration file or on the command line.

Chat scripts are not executable, but the program that is invoked by `connect` must be executable. You might use the chat utility as the program to be invoked by `connect`. In this instance, if you store the chat script in an external file through the `-f` option, then your chat script file is not executable.

The `chat` program that is described in `chat(1m)` executes the actual chat script. The `pppd` daemon invokes the `chat` program whenever `pppd` encounters the `connect "chat ..."` option.

Note – You can use any external program, such as `Perl` or `Tcl`, to create advanced chat scripts. The `chat` utility is provided as a convenience.

▼ How to Invoke a Chat Script (Task)

1. **Create the chat script as an ASCII file.**
2. **Invoke the chat script in any PPP configuration file by using the following syntax:**

   ```
   connect 'chat  -f /etc/ppp/chatfile'
   ```

   The `-f` flag indicates that a file name is to follow. */etc/ppp/chatfile* represents the name of the chat file.
3. **Give read permission for the external chat file to the user who runs the `pppd` command.**

Caution – The chat program always runs with the user's privileges, even if the `connect 'chat ...'` option is invoked from a privileged source. Thus, a separate chat file that is read with the `-f` option must be readable by the invoking user. This privilege can be a security problem if the chat script contains passwords or other sensitive information.

Example 8–1 Inline Chat Script

You can place the entire chat script conversation on a single line, similar to the following:

```
connect 'chat "" "AT&F1" OK ATDT5551212 CONNECT "\c"'
```

The complete chat script follows the `chat` keyword. The script terminates with `"\c"'`. You use this form in any PPP configuration file or on the command line as an argument to `pppd`.

More Information Chat Script in an External File

If the chat script that is needed for a particular peer is long or complicated, consider creating the script as a separate file. External chat files are easy to maintain and to document. You can add comments to the chat file by preceding the comments with the hash (#) sign.

The procedure "How to Create the Instructions for Calling a Peer" on page 55 shows the use of a chat script that is contained in an external file.

Creating a Chat File That Is Executable

You can create a chat file that is an executable script to be run automatically when the dial-up link is initiated. Thus, you can run additional commands during link initiation, such as `stty` for parity settings, besides the commands that are contained in a traditional chat script.

This executable chat script logs in to an old-style UNIX system that requires 7 bits with even parity. The system then changes to 8 bits with no parity when running PPP.

```
#!/bin/sh
chat "" "AT&F1" OK "ATDT555-1212" CONNECT "\c"
stty evenp
chat ogin: pppuser ssword: "\q\U" % "exec pppd"
stty -evenp
```

▼ How to Create an Executable Chat Program

1. **Use your text editor to create an executable chat program, such as the previous example.**
2. **Make the chat program executable.**

   `# chmod +x /etc/ppp/`*`chatprogram`*
3. **Invoke the chat program.**

   `connect /etc/ppp/`*`chatprogram`*

   Chat programs do not have to be located within the `/etc/ppp` file system. You can store chat programs in any location.

Authenticating Callers on a Link

This section explains how the PPP authentication protocols work and explains the databases that are associated with the authentication protocols.

Password Authentication Protocol (PAP)

PAP authentication is somewhat similar in operation to the UNIX `login` program, though PAP does not grant shell access to the user. PAP uses the PPP configuration files and PAP database in the form of the `/etc/ppp/pap-secrets` file for setting up authentication. PAP also uses `/etc/ppp/pap-secrets` for defining PAP security credentials. These credentials include a peer name, a "user name" in PAP parlance, and a password. PAP credentials also contain related information for each caller who is permitted to link to the local machine. The PAP user names and passwords can be identical to or different from the UNIX user names and passwords in the password database.

`/etc/ppp/pap-secrets` File

The PAP database is implemented in the `/etc/ppp/pap-secrets` file. Machines on both sides of the PPP link must have properly configured PAP credentials in their `/etc/ppp/pap-secrets` files for successful authentication. The caller (authenticatee) supplies credentials in the `user` and `password` columns of the `/etc/ppp/pap-secrets` file or in the obsolete `+ua` file. The server (authenticator) validates these credentials against information in `/etc/ppp/pap-secrets`, through the UNIX `passwd` database, or in the PAM facility.

The `/etc/ppp/pap-secrets` file has the following syntax.

```
myclient ISP-server mypassword *
```

The parameters have the following meaning.

`myclient`	PAP user name of the caller. Often, this name is identical to the caller's UNIX user name, particularly if the dial-in server uses the `login` option of PAP.
`ISP-server`	Name of the remote machine, often a dial-in server.
`mypassword`	Caller's PAP password.
*	IP address that is associated with the caller. Use an asterisk (*) to indicate any IP address.

Creating PAP Passwords

PAP passwords are sent over the link *in the clear*, that is, in readable ASCII format. For the caller (authenticatee), the PAP password must be stored in the clear in any of the following locations:

- In `/etc/ppp/pap-secrets`
- In another external file
- In a named pipe through the `pap-secrets` @ feature
- As an option to `pppd`, either on the command line or in a PPP configuration file
- Through the `+ua` file

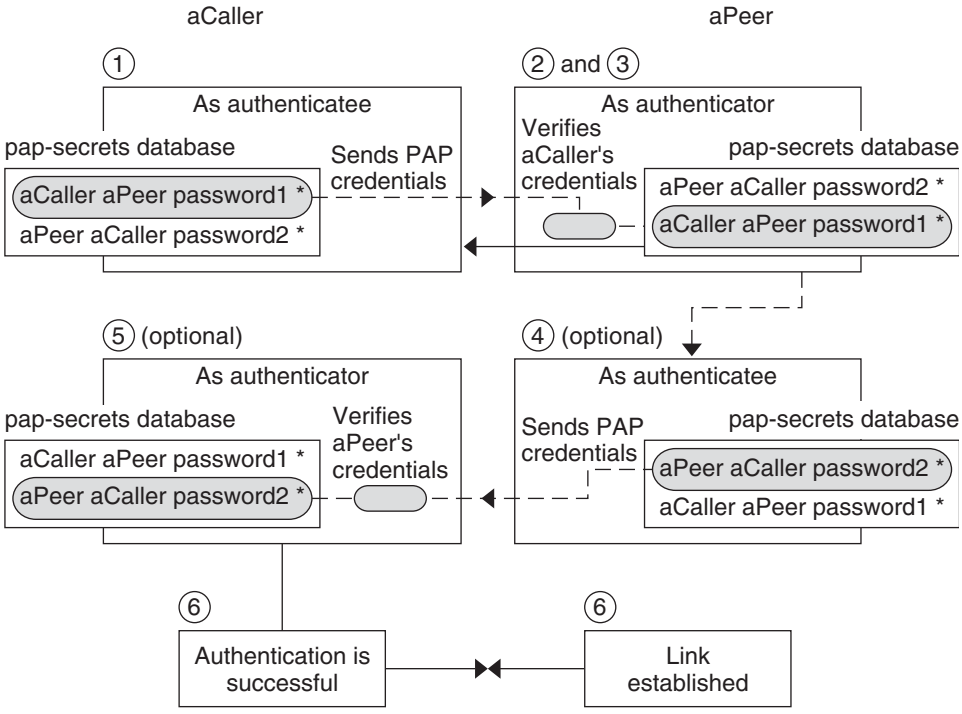
On the server (authenticator), the PAP password can be hidden by doing one of the following:

- Specifying `papcrypt` and using passwords that are hashed by `crypt`(3C) in the `pap-secrets` file.
- Specifying the `login` option to `pppd` and omitting the password from the `pap-secrets` file by placing double quotes (`""`) in the password column. In this instance, authentication is performed through the UNIX passwd database or the PAM mechanism.

What Happens During PAP Authentication

PAP authentication occurs in the following sequence.

FIGURE 8–1 PAP Authentication Process



1. The caller (authenticatee) calls the remote peer (authenticator) and provides its PAP user name and password as part of link negotiation.
2. The peer verifies the identity of the caller in its `/etc/ppp/pap-secrets` file. If the peer uses the `login` option of PAP, the peer verifies the caller's user name and password in its password database.
3. If authentication is successful, the peer continues link negotiation with the caller. If authentication fails, the link is dropped.
4. (Optional) If the caller authenticates responses from remote peers, the remote peer must send its own PAP credentials to the caller. Thus, the remote peer becomes the authenticatee and the caller the authenticator.
5. (Optional) The original caller reads its own `/etc/ppp/pap-secrets` to verify the identity of the remote peer.

Note – If the original caller does require authentication credentials from the remote peer, Step 1 and Step 4 happen in parallel.

If the peer is authenticated, negotiation continues. Otherwise, the link is dropped.

6. Negotiation between caller and peer continues until the link is successfully established.

Using the `login` Option With `/etc/ppp/pap-secrets`

You can add the `login` option for authenticating PAP credentials to any PPP configuration file. When `login` is specified, for example, in `/etc/ppp/options`, `pppd` verifies that the caller's PAP credentials exist in the password database. The following shows the format of a `/etc/ppp/pap-secrets` file with the `login` option.

```
joe    *  ""  *
sally  *  ""  *
sue    *  ""  *
```

The parameters have the following meanings.

Caller	`joe`, `sally`, and `sue` are the names of the authorized callers.
Server	Asterisk (*), which indicates that any server name is valid. The `name` option is not required in the PPP configuration files.
Password	Double quotes, which indicate that any password is valid. If a password is in this column, then the password from the peer must match both the PAP password and the UNIX `passwd` database.
IP Addresses	Asterisk (*), which indicates that any IP address is allowed.

Challenge-Handshake Authentication Protocol (CHAP)

CHAP authentication uses the notion of the *challenge* and *response*, which means that the peer (authenticator) challenges the caller (authenticatee) to prove its identity. The challenge includes a random number and a unique ID that is generated by the authenticator. The caller must use the ID, random number, and its CHAP security credentials to generate the proper response (handshake) to send to the peer.

CHAP security credentials include a CHAP user name and a CHAP "secret." The CHAP secret is an arbitrary string that is known to both the caller and the peer before they negotiate a PPP link. You configure CHAP security credentials in the CHAP database, `/etc/ppp/chap-secrets`.

`/etc/ppp/chap-secrets` File

The CHAP database is implemented in the `/etc/ppp/chap-secrets` file. Machines on both sides of the PPP link must have each others' CHAP credentials in their `/etc/ppp/chap-secrets` files for successful authentication.

Note – Unlike PAP, the shared secret must be in the clear on both peers. You cannot use crypt, PAM, or the PPP login option with CHAP.

The `/etc/ppp/chap-secrets` file has the following syntax.

```
myclient myserver secret5748 *
```

The parameters have the following meanings:

`myclient` CHAP user name of the caller. This name can be the same as or different from the caller's UNIX user name.

`myserver` Name of the remote machine, often a dial-in server.

`secret5748` Caller's CHAP secret.

Note – Unlike PAP passwords, CHAP secrets are never sent over the link. Rather, CHAP secrets are used when the local machines compute the response.

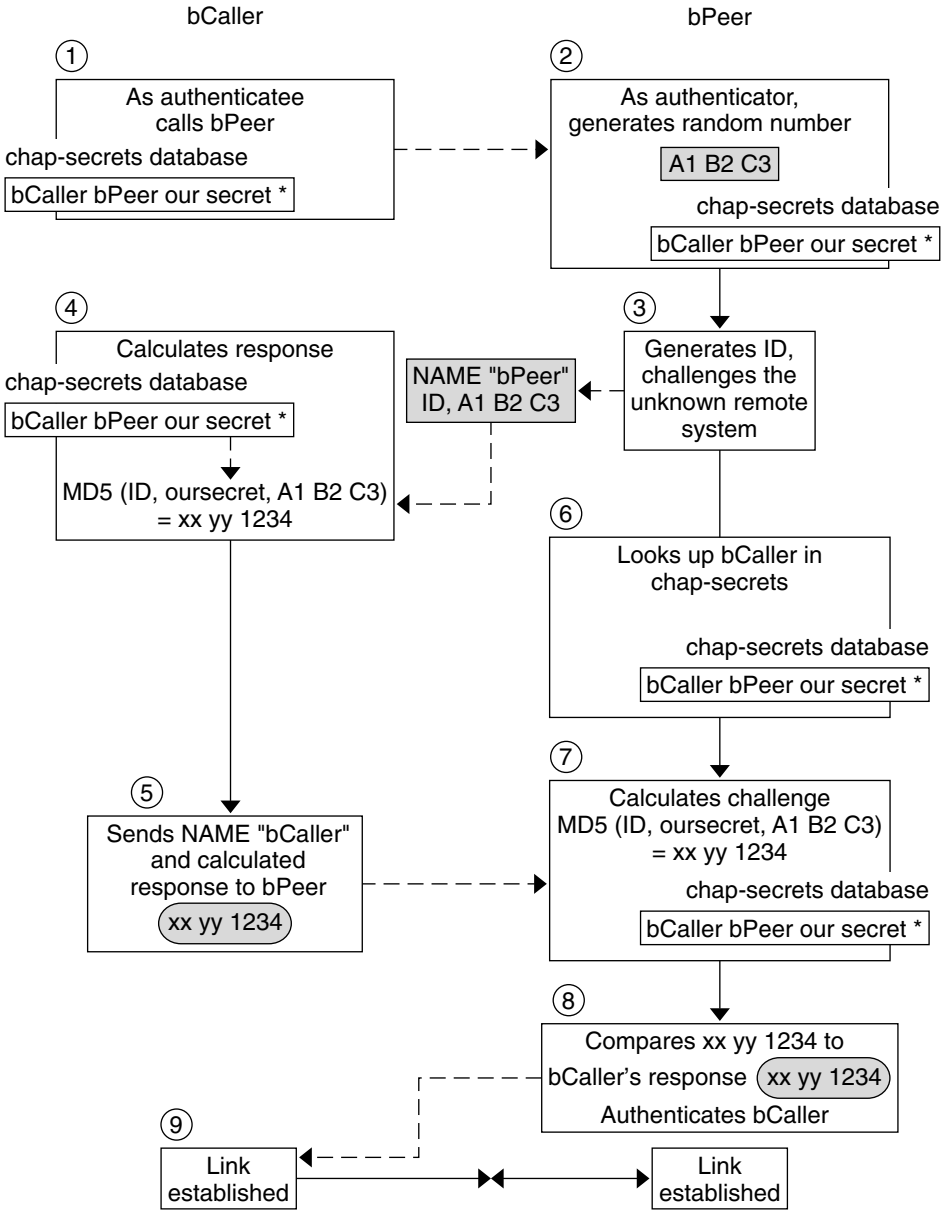
---

* IP address that is associated with the caller. Use an asterisk (*) to indicate any IP address.

What Happens During CHAP Authentication

CHAP authentication occurs in the following sequence.

FIGURE 8–2 CHAP Authentication Sequence



1. Two peers that are about to initiate communications agree on a secret to be used for authentication during negotiation of a PPP link.

2. The administrators of both machines add the secret, CHAP user names, and other CHAP credentials to the `/etc/ppp/chap-secrets` database of their respective machines.
3. The caller (authenticatee) calls the remote peer (authenticator).
4. The authenticator generates a random number and an ID, and sends this data to the authenticatee as a challenge.
5. The authenticatee looks up the peer's name and secret in its `/etc/ppp/chap-secrets` database.
6. The authenticatee calculates a response by applying the MD5 computational algorithm to the secret and the peer's random number challenge. Then the authenticatee sends the results as its response to the authenticator.
7. The authenticator looks up the authenticatee's name and secret in its `/etc/ppp/chap-secrets` database.
8. The authenticator calculates its own figure by applying MD5 to the number that was generated as the challenge and the secret for the authenticatee in `/etc/ppp/chap-secrets`.
9. The authenticator compares its results with the response from the caller. If the two numbers are the same, the peer has successfully authenticated the caller, and link negotiation continues. Otherwise the link is dropped.

Creating an IP Addressing Scheme for Callers

Consider creating one or more IP addresses for all incoming calls instead of assigning a unique IP address to each remote user. Dedicated IP addresses are particularly important if the number of potential callers exceeds the number of serial ports and modems on the dial-in server. You can implement a number of different scenarios, depending on your site's needs. Moreover, the scenarios are not mutually exclusive.

Assigning Dynamic IP Addresses to Callers

Dynamic addressing involves the assignment to each caller of the IP address that is defined in `/etc/ppp/options.`*ttyname*. Dynamic addressing occurs on a per-serial port basis. When a call arrives over a serial line, the caller receives the IP address in the `/etc/ppp/options.`*ttyname* file for the call's serial interface.

For example, suppose a dial-in server has four serial interfaces that provide dial-up service to incoming calls:

- For serial port `term/a`, create the file `/etc/ppp/options.term.a` with the following entry:

  ```
  :10.1.1.1
  ```

- For serial port `term/b`, create the file `/etc/ppp/options.term.b` with the following entry:

```
:10.1.1.2
```

- For serial port `term/c`, create the file `/etc/ppp/options.term.c` with the following entry:

```
:10.1.1.3
```

- For serial port `term/d`, create the file `/etc/ppp/options.term.d` with the following entry:

```
:10.1.1.4
```

With the previous addressing scheme, an incoming call on serial interface `/dev/term/c` is given the IP address 10.1.1.3 for the duration of the call. After the first caller hangs up, a later call that comes in over serial interface `/dev/term/c` is also given the IP address 10.1.1.3.

The advantages of dynamic addressing include the following:

- You can track PPP network usage down to the serial port.
- You can assign a minimum number of IP addresses for PPP use.
- You can administer IP filtering in a more simplified fashion.

Assigning Static IP Addresses to Callers

If your site implements PPP authentication, you can assign specific, *static* IP addresses to individual callers. In this scenario, every time a dial-out machine calls the dial-in server, the caller receives the same IP address.

You implement static addresses in either the pap-secrets or chap-secrets database. Here is a example of an `/etc/ppp/pap-secrets` file that defines static IP addresses.

```
joe    myserver  joepasswd   10.10.111.240
sally  myserver  sallypasswd 10.10.111.241
sue    myserver  suepasswd   10.10.111.242
```

Caller	`joe`, `sally`, and `sue` are the names of the authorized callers.
Server	`myserver` indicates the name of the server.
Password	`joepasswd`, `sallypasswd`, and `suepasswd` indicate the passwords for each caller.
IP Addresses	`10.10.111.240` and `10.10.111.241` and `10.10.111.242` are the IP addresses assigned to each caller.

Here is a example of an `/etc/ppp/chap-secrets` file that defines static IP addresses.

```
account1 myserver secret5748  10.10.111.244
account2 myserver secret91011 10.10.111.245
```

Caller	`account1` and `account2` indicate the names of the callers.
Server	`myserver` indicates the name of the server for each caller.

Password	`secret5748` and `secret91011` indicates the CHAP secret for each caller.
IP Addresses	`10.10.111.244` and `10.10.111.245` are the IP addresses for each caller.

Assigning IP Addresses by sppp Unit Number

If you are using either PAP or CHAP authentication, you can assign IP addresses to callers by the `sppp` unit number. The following shows an example of this usage.

```
myclient ISP-server mypassword 10.10.111.240/28+
```

The plus sign (+) indicates that the unit number is added to the IP address. Note the following:

- Addresses `10.10.111.240` through `10.10.111.255` are assigned to remote users.
- `sppp0` gets IP address `10.10.111.240`.
- `sppp1` gets IP address `10.10.111.241` and so on.

Creating PPPoE Tunnels for DSL Support

By using PPPoE, you can provide PPP over high-speed digital services to multiple clients that are using one or more DSL modems. PPPoE implements these services by creating an Ethernet tunnel through three participants: the enterprise, the telephone company, and the service provider.

- For an overview and description of how PPPoE works, see "PPPoE Overview" on page 31.
- For tasks for setting up PPPoE tunnels, see Chapter 6, "Setting Up a PPPoE Tunnel (Tasks)."

This section contains detailed information about PPPoE commands and files, which is summarized in the next table.

TABLE 8–2 PPPoE Commands and Configuration Files

File or Command	Description	For Instructions
`/etc/ppp/pppoe`	A file that contains characteristics that are applied by default to all tunnels that were set up by PPPoE on the system	"`/etc/ppp/pppoe` File" on page 142
`/etc/ppp/pppoe.`*device*	A file that contains characteristics of a particular interface that is used by PPPoE for a tunnel	"`/etc/ppp/pppoe.`*device* File" on page 144
`/etc/ppp/pppoe.if`	File that lists the Ethernet interface over which runs the tunnel that is set up by PPPoE	"`/etc/ppp/pppoe.if` File" on page 140
`/usr/sbin/sppptun`	Command for configuring the Ethernet interfaces that are involved in a PPPoE tunnel	"`/usr/sbin/sppptun` Command" on page 140

TABLE 8–2 PPPoE Commands and Configuration Files *(Continued)*

File or Command	Description	For Instructions
`/usr/lib/inet/pppoed`	Command and options for using PPPoE to set up a tunnel	"`/usr/lib/inet/pppoed` Daemon" on page 142

Files for Configuring Interfaces for PPPoE

The interfaces that are used at either end of the PPPoE tunnel must be configured before the tunnel can support PPP communications. Use `/usr/sbin/sppptun` and `/etc/ppp/pppoe.if` files for this purpose. You must use these tools to configure Ethernet interfaces on all Oracle Solaris PPPoE clients and PPPoE access servers.

`/etc/ppp/pppoe.if` File

The `/etc/ppp/pppoe.if` file lists the names of all Ethernet interfaces on a host to be used for the PPPoE tunnels. This file is processed during system boot when the interfaces that are listed are plumbed for use in PPPoE tunnels.

You need to create explicitly `/etc/ppp/pppoe.if`. Type the name of one interface to be configured for PPPoE on each line.

The following example shows an `/etc/ppp/pppoe.if` file for a server that offers three interfaces for PPPoE tunnels.

```
# cat /etc/ppp/pppoe.if
hme1
hme2
hme3
```

PPPoE clients usually have only one interface that is listed in `/etc/ppp/pppoe.if`.

`/usr/sbin/sppptun` Command

You can use the `/usr/sbin/sppptun` command to manually plumb and unplumb the Ethernet interfaces to be used for PPPoE tunnels. By contrast, `/etc/ppp/pppoe.if` is only read when the system boots. These interfaces should correspond to the interfaces that are listed in `/etc/ppp/pppoe.if`.

`sppptun` plumbs the Ethernet interfaces that are used in PPPoE tunnels in a manner that is similar to the `ipadm` command. Unlike `ipadm`, you must plumb interfaces twice to support PPPoE because two Ethernet protocol numbers are involved.

The basic syntax for `sppptun` is as follows:

```
# /usr/sbin/sppptun plumb pppoed device-name
      device-name:pppoed
# /usr/sbin/sppptun plumb pppoe device-name
      device-name:pppoe
```

In this syntax, *device-name* is the name of the device to be plumbed for PPPoE.

The first time that you issue the `sppptun` command, the discovery protocol `pppoed` is plumbed on the interface. The second time that you run `sppptun`, the session protocol `pppoe` is plumbed. `sppptun` prints the name of the interface that was just plumbed. You use this name to unplumb the interface, when necessary.

For more information, refer to the `sppptun(1M)` man page.

Examples of `sppptun` Commands for Administering Interfaces

The following example shows how to manually plumb an interface for PPPoE by using `/usr/sbin/sppptun`.

```
# /usr/sbin/sppptun plumb pppoed hme0
hme0:pppoed
# /dev/sppptun plumb pppoe hme0
 hme0:pppoe
```

This example shows how to list the interfaces on an access server that was plumbed for PPPoE.

```
# /usr/sbin/sppptun query
hme0:pppoe
hme0:pppoed
hme1:pppoe
hme1:pppoed
hme2:pppoe
hme2:pppoed
```

This example shows how to unplumb an interface.

```
# sppptun unplumb hme0:pppoed
# sppptun unplumb hme0:pppoe
```

PPPoE Access Server Commands and Files

A service provider that offers DSL services or support to customers can use an access server that is running PPPoE. The PPPoE access server and client do function in the traditional client-server relationship. This relationship is similar to the relationship of the dial-out machine and dial-in server on a dial-up link. One PPPoE system initiates communications and one PPPoE system answers. By contrast, the PPP protocol has no notion of the client-server relationship. PPP considers both systems equal peers.

The commands and files that set up a PPPoE access server include the following:

- "`/usr/sbin/sppptun` Command" on page 140
- "`/usr/lib/inet/pppoed` Daemon" on page 142
- "`/etc/ppp/pppoe` File" on page 142
- "`/etc/ppp/pppoe.`*device* File" on page 144

- "pppoe.so Shared Object" on page 147

/usr/lib/inet/pppoed Daemon

The pppoed daemon accepts broadcasts for services from prospective PPPoE clients. Additionally, pppoed negotiates the server side of the PPPoE tunnel and runs pppd, the PPP daemon, over that tunnel.

You configure pppoed services in the /etc/ppp/pppoe and /etc/ppp/pppoe.*device* files. If /etc/ppp/pppoe exists when the system boots, pppoed runs automatically. You can also explicitly run the pppoed daemon on the command line by typing /usr/lib/inet/pppoed.

/etc/ppp/pppoe File

The /etc/ppp/pppoe file describes the services that are offered by an access server plus options that define how PPP runs over the PPPoE tunnel. You can define services for individual interfaces, or globally, that is, for all interfaces on the access server. The access server sends the information in the /etc/ppp/pppoe file in response to a broadcast from a potential PPPoE client.

The following is the basic syntax of /etc/ppp/pppoe:

global-options
service service-name
    service-specific-options
    device interface-name

The parameters have the following meanings.

global-options
: Sets the default options for the /etc/ppp/pppoe file. These options can be any options that are available through pppoed or pppd. For complete lists of options, see the man pages pppoed(1M) and pppd(1M).

  For example, you must list the Ethernet interfaces that are available for the PPPoE tunnel as part of *global options*. If you do not define devices in /etc/ppp/pppoe, the services are not offered on any interface.

  To define devices as a global option, use the following form:

  device interface <,*interface*>

  interface specifies the interface where the service listens for potential PPPoE clients. If more than one interface is associated with the service, separate each name with a comma.

service service-name	Starts the definition of the service *service-name*. *service-name* is a string that can be any phrase that is appropriate to the services that are provided.
service-specific-options	Lists the PPPoE and PPP options specific to this service.
`device` *interface-name*	Specifies the interface where the previously listed service is available.

For additional options to `/etc/ppp/pppoe`, refer to the `pppoed`(1M) and `pppd`(1M) man pages.

A typical `/etc/ppp/pppoe` file might resemble the following.

EXAMPLE 8–2 Basic `/etc/ppp/pppoe` File

```
device hme1,hme2,hme3
service internet
   pppd "name internet-server"
service intranet
   pppd "192.168.1.1:"
service debug
   device hme1
   pppd "debug name internet-server"
```

In this file, the following values apply.

`hme1,hme2,hme3`	Three interfaces on the access server to be used for PPPoE tunnels.
`service internet`	Advertises a service that is called `internet` to prospective clients. The provider that offers the service also determines how `internet` is defined. For example, a provider might interpret `internet` to mean various IP services, as well as access to the Internet.
`pppd`	Sets the command-line options that are used when the caller invokes `pppd`. The option "`name internet-server`" gives the name of the local machine, the access server, as `internet-server`.
`service intranet`	Advertises another service that is called `intranet` to prospective clients.
`pppd "192.168.1.1:"`	Sets the command-line options that are used when the caller invokes `pppd`. When the caller invokes `pppd`, 192.168.1.1 is set as the IP address for the local machine, the access server.
`service debug`	Advertises a third service, debugging, on the interfaces that are defined for PPPoE.

`device hme1`	Restricts debugging to PPPoE tunnels to `hme1`.
`pppd "debug name internet-server"`	Sets the command-line options that are used when the caller invokes `pppd`, in this instance, PPP debugging on `internet-server`, the local machine.

`/etc/ppp/pppoe.`*device* File

The `/etc/ppp/pppoe.`*device* file describes the services that are offered on one interface of a PPPoE access server. `/etc/ppp/pppoe.`*device* also includes options that define how PPP runs over the PPPoE tunnel. `/etc/ppp/pppoe.`*device* is an optional file, which operates exactly like the global `/etc/ppp/pppoe`. However, if `/etc/ppp/pppoe.`*device* is defined for an interface, its parameters have precedence for that interface over the global parameters that are defined in `/etc/ppp/pppoe`.

The basic syntax of `/etc/ppp/pppoe.`*device* is as follows:

```
service service-name
      service-specific-options
service another-service-name
       service-specific-options
```

The only difference between this syntax and the syntax of `/etc/ppp/pppoe` is that you cannot use the `device` option that is shown in "`/etc/ppp/pppoe` File" on page 142.

`pppoe.so` Plugin

`pppoe.so` is the PPPoE shared object file that must be invoked by PPPoE access servers and clients. This file limits MTU and MRU to 1492, filters packets from the driver, and negotiates the PPPoE tunnel, along with `pppoed`. On the access server side, `pppoe.so` is automatically invoked by the `pppd` daemon.

Using PPPoE and PPP Files to Configure an Access Server

This section contains samples of all files that are used to configure an access server. The access server is multihomed. The server is attached to three subnets: `green`, `orange`, and `purple`. `pppoed` runs as `root` on the server, which is the default.

PPPoE clients can access the `orange` and `purple` networks through interfaces `hme0` and `hme1`. Clients log in to the server by using the standard UNIX login. The server authenticates the clients by using PAP.

The `green` network is not advertised to clients. The only way clients can access `green` is by directly specifying "`green-net`" and supplying CHAP authentication credentials. Moreover, only clients `joe` and `mary` are allowed to access the `green` network by using static IP addresses.

EXAMPLE 8–3 `/etc/ppp/pppoe` File for an Access Server

```
service orange-net
     device hme0,hme1
     pppd "require-pap login name orange-server orange-server:"
service purple-net
     device hme0,hme1
     pppd "require-pap login name purple-server purple-server:"
service green-net
     device hme1
     pppd "require-chap name green-server green-server:"
     nowildcard
```

This sample describes the services that are available from the access server. The first service section describes the services of the `orange` network.

```
service orange-net
     device hme0,hme1
     pppd "require-pap login name orange-server orange-server:"
```

Clients access the `orange` network over interfaces `hme0` and `hme1`. The options that are given to the `pppd` command force the server to require PAP credentials from potential clients. The `pppd` options also set the server's name to `orange-server`, as used in the `pap-secrets` file.

The service section for the `purple` network is identical to the service section of the `orange` network except for the network and server names.

The next section describes the services of the `green` network:

```
service green-net
     device hme1
     pppd "require-chap name green-server green-server:"
     nowildcard
```

This section restricts client access to interface `hme1`. Options that are given to the `pppd` command force the server to require CHAP credentials from prospective clients. The `pppd` options also set the server name to `green-server`, to be used in the `chap-secrets` file. The `nowildcard` option specifies that the existence of the green network is not advertised to clients.

For this access server scenario just discussed, you might set up the following `/etc/ppp/options` file.

EXAMPLE 8–4 `/etc/ppp/options` File for an Access Server

```
     auth
     proxyarp
     nodefaultroute
     name no-service    # don’t authenticate otherwise
```

The option `name no-service` overrides the server name that is normally searched for during PAP or CHAP authentication. The server's default name is the one found by the `/usr/bin/hostname` command. The `name` option in the previous example changes the server's name to `no-service`. The name `no-service` is not likely to be found in a `pap` or `chap-secrets` file. This action prevents a random user from running `pppd` and overriding the `auth` and `name` options that are set in `/etc/ppp/options`. `pppd` then fails because no secrets can be found for the client with a server name of `no-service`.

The access server scenario uses the following `/etc/hosts` file.

EXAMPLE 8–5 `/etc/hosts` File for an Access Server

```
     172.16.0.1    orange-server
     172.17.0.1    purple-server
     172.18.0.1    green-server
     172.18.0.2    joes-pc
     172.18.0.3    marys-pc
```

Here is the `/etc/ppp/pap-secrets` file that is used for PAP authentication for clients that attempt to access the `orange` and `purple` networks.

EXAMPLE 8–6 `/etc/ppp/pap-secrets` File for an Access Server

```
* orange-server "" 172.16.0.2/16+
* purple-server "" 172.17.0.2/16+
```

Here is the `/etc/ppp/chap-secrets` file that is used for CHAP authentication. Note that only clients `joe` and `mary` are listed in the file.

EXAMPLE 8–7 `/etc/ppp/chap-secrets` File for an Access Server

```
 joe green-server "joe's secret" joes-pc
mary green-server "mary's secret" marys-pc
```

PPPoE Client Commands and Files

To run PPP over a DSL modem, a machine must become a PPPoE client. You have to plumb an interface to run PPPoE, and then use the `pppoec` utility to "discover" the existence of an access server. Thereafter, the client can create the PPPoE tunnel over the DSL modem and run PPP.

The PPPoE client relates to the access server in the traditional client-server model. The PPPoE tunnel is not a dial-up link, but the tunnel is configured and operated in much the same manner.

The commands and files that set up a PPPoE client include the following:

- "`/usr/sbin/sppptun` Command" on page 140
- "`/usr/lib/inet/pppoec` Utility" on page 147
- "`pppoe.so` Shared Object" on page 147
- "`/etc/ppp/peers/`*peer-name* File" on page 120
- "`/etc/ppp/options` Configuration File" on page 115

`/usr/lib/inet/pppoec` Utility

The `/usr/lib/inet/pppoec` utility is responsible for negotiating the client side of a PPPoE tunnel. `pppoec` is similar to the `chat` utility. You do not invoke `pppoec` directly. Rather, you start `/usr/lib/inet/pppoec` as an argument to the `connect` option of `pppd`.

`pppoe.so` Shared Object

`pppoe.so` is the PPPoE shared object that must be loaded by PPPoE to provide PPPoE capability to access servers and clients. The `pppoe.so` shared object limits MTU and MRU to 1492, filters packets from the driver, and handles runtime PPPoE messages.

On the client side, `pppd` loads `pppoe.so` when the user specifies the `plugin pppoe.so` option.

`/etc/ppp/peers/`*peer-name* File for Defining an Access Server Peer

When you define an access server to be discovered by `pppoec`, you use options that apply to both `pppoec` and the `pppd` daemon. An `/etc/ppp/peers/`*peer-name* file for an access server requires the following parameters:

- `sppptun` – Name for the serial device that is used by the PPPoE tunnel.
- `plugin pppoe.so` – Instructs `pppd` to load the `pppoe.so` shared object.
- `connect "/usr/lib/inet/pppoec` *device*" – Starts a connection. `connect` then invokes the `pppoec` utility over *device*, the interface that is plumbed for PPPoE.

The remaining parameters in the `/etc/ppp/peers/`*peer-name* file should apply to the PPP link on the server. Use the same options that you would for `/etc/ppp/peers/`*peer-name* on a dial-out machine. Try to limit the number of options to the minimum you need for the PPP link.

The following example is introduced in "How to Define a PPPoE Access Server Peer" on page 87.

EXAMPLE 8–8 `/etc/ppp/peers/`*peer-name* to Define a Remote Access Server

```
# cat /etc/ppp/peers/dslserve
sppptun
plugin pppoe.so
connect "/usr/lib/inet/pppoec hme0"
```

EXAMPLE 8–8 `/etc/ppp/peers/`*peer-name* to Define a Remote Access Server *(Continued)*

```
noccp
noauth
user Red
password redsecret
noipdefault
defaultroute
```

This file defines parameters to be used when setting up a PPPoE tunnel and PPP link to access server `dslserve`. The options that are included are as follows.

Option	Description
`sppptun`	Defines `sppptun` as the name of the serial device.
`plugin pppoe.so`	Instructs `pppd` to load the `pppoe.so` shared object.
`connect "/usr/lib/inet/pppoec hme0"`	Runs `pppoec` and designates `hme0` as the interface for the PPPoE tunnel and PPP link.
`noccp`	Turns off CCP compression on the link. **Note –** Many ISPs use only proprietary compression algorithms. Turning off the publicly available CCP algorithm saves negotiation time and avoids very occasional interoperability problems.
`noauth`	Stops `pppd` from demanding authentication credentials from the access server. Most ISPs do not provide authentication credentials to customers.
`user Red`	Sets the name `Red` as the user name for the client, which is required for PAP authentication by the access server.
`password redsecret`	Defines `redsecret` as the password to be provided to the access server for PAP authentication.
`noipdefault`	Assigns 0.0.0.0 as the initial IP address.
`defaultroute`	Tells `pppd` to install a default IPv4 route after IPCP negotiation. You should include `defaultroute` in `/etc/ppp/peers/`*peer-name* when the link is the system's link to the Internet, which is true for a PPPoE client.

◆ ◆ ◆ CHAPTER 9

Migrating From Asynchronous Solaris PPP to Solaris PPP 4.0 (Tasks)

Earlier versions of the Oracle Solaris OS included a different PPP implementation, Asynchronous Solaris PPP (`asppp`). If you want to convert peers that run `asppp` to the newer PPP 4.0, you need to run a conversion script. This chapter covers the following topics in PPP conversion:

The chapter uses a sample `asppp` configuration to explain how to accomplish PPP conversion. For a description of the differences between Solaris PPP 4.0 and `asppp`, go to "Which Version of Solaris PPP to Use" on page 20.

Before Converting `asppp` Files

You can use the conversion script `/usr/sbin/asppp2pppd` to convert the files that compose a standard `asppp` configuration:

- `/etc/asppp.cf` – Asynchronous PPP configuration file
- `/etc/uucp/Systems` – UUCP file that describes the characteristics of the remote peer
- `/etc/uucp/Devices` – UUCP file that describes the modem on the local machine
- `/etc/uucp/Dialers` – UUCP file that contains the login sequence to be used by the modem that is described in the `/etc/uucp/Devices` file

For more information about `asppp`, see the *Solaris 8 System Administration Collection, Volume 3*, available from `http://docs.sun.com`.

Example of the `/etc/asppp.cf` Configuration File

The procedure that is shown in "How to Convert From `asppp` to Solaris PPP 4.0" on page 152 uses the following `/etc/asppp.cf` file.

```
#
ipadm create-if ipdptp0
ipadm create-addr -T static -a local=mojave,remote=gobi ipdptp0/ppaddr

path
 inactivity_timeout 120     # Approx. 2 minutes
    interface ipdptp0
    peer_system_name Pgobi    # The name we log in with (also in
                                    # /etc/uucp/Systems
```

The file contains the following parameters.

`ifpadm create-if ipdptp0`
: Runs the `ipadm` command to create an interface called `ipdptp0`

`ipadm create-addr -T static -a local=mojave,remote=gobi ipdptp0/ppadd`
: Runs the `ipadm` command to configure a link from PPP interface `ipdptp0` on the local machine `mojave` to the remote peer `gobi`

`inactivity_timeout 120`
: Terminates the line after two minutes of inactivity

`interface ipdptp0`
: Configures the interface `ipdptp0` on the dial-out machine for asynchronous PPP

`peer_system_name Pgobi`
: Gives the name of the remote peer, `Pgobi`

Example of the `/etc/uucp/Systems` File

The procedure that is shown in "How to Convert From asppp to Solaris PPP 4.0" on page 152 uses the following `/etc/uucp/Systems` file.

```
#ident  "@(#)Systems    1.5  92/07/14 SMI"  /* from SVR4 bnu:Systems 2.4 */
#

# .
# .
Pgobi Any ACU 38400 15551212 in:--in: mojave word: sand
```

The file contains the following parameters:

`Pgobi`	Uses `Pgobi` as the host name of the remote peer.
`Any ACU`	Tells the modem on the dial-out machine `mojave` to establish a link with a modem on `Pgobi` at any time of the day. Any ACU means "look for ACU in the `/etc/uucp/Devices` file."
`38400`	Sets 38400 as the maximum speed of the link.
`15551212`	Gives the telephone number of `Pgobi`.

`in:–in: mojave word: sand`	Defines the login script that is required by `Pgobi` to authenticate dial-out machine `mojave`.

Example of the `/etc/uucp/Devices` File

The procedure that is shown in "How to Convert From asppp to Solaris PPP 4.0" on page 152 uses the following `/etc/uucp/Devices` file.

```
#ident  "@(#)Devices  1.6  92/07/14 SMI"  /* from SVR4 bnu:Devices 2.7 */

.
.
#

TCP,et - - Any TCP -
.
.
#
ACU cua/b - Any hayes
# 0-7 are on a Magma 8 port card
Direct cua/0 - Any direct
Direct cua/1 - Any direct
Direct cua/2 - Any direct
Direct cua/3 - Any direct
Direct cua/4 - Any direct
Direct cua/5 - Any direct
Direct cua/6 - Any direct
Direct cua/7 - Any direct
# a is the console port (aka "tip" line)
Direct cua/a - Any direct
# b is the aux port on the motherboard
Direct cua/b - Any direct
# c and d are high speed sync/async ports
Direct cua/c - Any direct
Direct cua/d - Any direct
```

The file supports any Hayes modem that is connected to serial port `cua/b`.

Example of the `/etc/uucp/Dialers` File

The procedure that is shown in "How to Convert From asppp to Solaris PPP 4.0" on page 152 uses the following `/etc/uucp/Dialers` file.

```
#
#       <Much information about modems supported by Oracle Solaris UUCP>


penril    =W-P    "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP > 9\c OK
ventel    =&-%    "" \r\p\r\c $ k\c ONLINE!
```

```
vadic    =K-K    "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE
develcon ""    "" \pr\ps\c est:\007 \E\D\e \n\007
micom    ""    "" \s\c NAME? \D\r\c GO
direct
#
#
#
#   Hayes Smartmodem -- modem should be set with the configuration
#   switches as follows:
#
#       S1 - UP        S2 - UP        S3 - DOWN    S4 - UP
#       S5 - UP        S6 - DOWN    S7 - ?        S8 - DOWN
#
hayes    =,-,    "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r \EATDT\T\r\c CONNECT
```

<much more information about modems supported by Oraccle Solaris UUCP>

This file contains the chat scripts for all types of modems, including the Hayes modems that are supported in the `/etc/uucp/Dialers` file.

Running the asppp2pppd Conversion Script (Tasks)

The `/usr/sbin/asppp2pppd` script copies the PPP information in `/etc/asppp.cf` and PPP-related UUCP files to appropriate locations in the Solaris PPP 4.0 files.

Task Prerequisites

Before doing the next task, you should have done the following:

- Installed the Oracle Solaris release on the machine that also has the `asppp` and UUCP configuration files
- Become superuser on the machine with the PPP files, for example, the machine `mojave`

▼ How to Convert From asppp to Solaris PPP 4.0

1 Start the conversion script.

```
# /usr/sbin/asppp2pppd
```

The conversion process starts and gives you the following screen output.

```
This script provides only a suggested translation for your existing aspppd
configuration.  You will need to evaluate for yourself whether the translation
```

```
is appropriate for your operating environment.
Continue [Yn]?
```

2 Type "Y" to continue.

You receive the following output.

```
Chat cannot do echo checking; requests for this removed.
Adding 'noauth' to /etc/ppp/options

Preparing to write out translated configuration:
    1 chat file:
        1.  /etc/ppp/chat.Pgobi.hayes
    2 option files:
        2.  /etc/ppp/peers/Pgobi
        3.  /etc/ppp/options
    1 script file:
        4.  /etc/ppp/demand
```

The new Solaris PPP 4.0 files have been generated.

How to View the Results of the Conversion

You can view the Solaris PPP 4.0 files that were created by the `/usr/sbin/asppp2pppd` conversion script at the end of the conversion process. The script displays the following list of options.

```
Enter option number:
        1 - view contents of file on standard output
        2 - view contents of file using /usr/bin/less
        3 - edit contents of file using /usr/bin/vi
        4 - delete/undelete file from list
        5 - rename file in list
        6 - show file list again
        7 - escape to shell (or "!")
        8 - abort without saving anything
        9 - save all files and exit (default)
Option:
```

1 Type 1 to view the contents of the files on the screen.

The script requests the number of the file you want to view.

```
File number (1 .. 4):
```

The numbers refer to the translated files that are listed during the conversion process, as shown in the previous Step 2.

2 Type 1 to view the chat file `/etc/ppp/chat.Pgobi.hayes`.

```
File number (1 .. 4):  1
"" \d\dA\p\pTE1V1X1Q0S2=255S12=255\r\c
OK\r ATDT\T\r\c
CONNECT \c
```

```
in:--in: mojave
word: sand
```

The chat script contains the modem "chat" information that appears on the hayes line in the sample `/etc/uucp/Dialers` file. `/etc/ppp/chat.Pgobi.hayes` also contains the login sequence for `Pgobi` that appears in the sample `/etc/uucp/Systems` file. The chat script is now in the `/etc/ppp/chat.Pgobi.hayes` file.

3 Type 2 to view the peers file, `/etc/ppp/peers/Pgobi`.

```
File number (1 .. 4):  2
/dev/cua/b
38400
demand
idle 120
connect "/usr/bin/chat -f /etc/ppp/chat.Pgobi.hayes -T '15551212'"
user NeverAuthenticate
mojave:gobi
```

The serial port information (`/dev/cua/b`) is from the `/etc/uucp/Devices` file. The link speed, idle time, authentication information, and peer names are from the `/etc/asppp.cf` file. "demand" refers to the "demand" script, to be called when the dial-out machine tries to connect to peer `Pgobi`.

4 Type 3 to view the `/etc/ppp/options` file that are created for dial-out machine `mojave`.

```
File number (1 .. 4):  3
#lock
noauth
```

The information in `/etc/ppp/options` is from the `/etc/asppp.cf` file.

5 Type 4 to view the contents of the `demand` script.

```
File number (1 .. 4):  4
/usr/bin/pppd file /etc/ppp/peers/Pgobi
```

This script, when invoked, runs the `pppd` command, which then reads the `/etc/ppp/peers/Pgobi` to initiate the link between `mojave` and `Pgobi`.

6 Type 9 to save the created files. Then exit the conversion script.

◆ ◆ ◆ CHAPTER 10

UUCP (Overview)

This chapter introduces the UNIX-to-UNIX Copy Program (UUCP) and its daemons. The following topics are covered:

UUCP enables computers to transfer files and exchange mail with each other. The program also enables computers to participate in large networks such as Usenet.

The Oracle Solaris OS provides the Basic Network Utilities (BNU) version of UUCP, also known as HoneyDanBer UUCP. The term *UUCP* denotes the complete range of files and utilities that compose the system, of which the program `uucp` is only a part. The UUCP utilities range from those utilities that are used to copy files between computers (`uucp` and `uuto`) to those utilities that are used for remote login and command execution (`cu` and `uux`).

UUCP Hardware Configurations

UUCP supports the following hardware configurations:

Direct links
: You can create a direct link to another computer by running RS-232 cables between serial ports on the two machines. Direct links are useful when two computers communicate regularly and are physically close, such as within 50 feet of each other. You can use a limited-distance modem to increase this distance somewhat.

Telephone lines
: By using an automatic call unit (ACU), such as a high-speed modem, your machine can communicate with other computers over standard phone lines. The modem dials the telephone number that is requested by UUCP. The recipient machine must have a modem that is capable of answering incoming calls.

Network — UUCP can also communicate over a network that runs TCP/IP or another protocol family. After your computer has been established as a host on a network, your computer can contact any other host that is connected to the network.

This chapter assumes that your UUCP hardware has already been assembled and configured. If you need to set up a modem, refer to the manuals that accompanied the modem for assistance.

UUCP Software

The UUCP software is automatically included when you run the Oracle Solaris installation program and select the entire distribution. Alternatively, you can add the UUCP software by using `pkgadd`. The UUCP programs can be divided into three categories: daemons, administrative programs, and user programs.

UUCP Daemons

The UUCP system has four daemons: `uucico`, `uuxqt`, `uusched`, and `in.uucpd`. These daemons handle UUCP file transfers and command executions. You can also run them manually from the shell, if necessary.

`uucico` — Selects the device that is used for the link, establishes the link to the remote computer, and performs the required login sequence and permission checks. Also, `uucico` transfers data files, execute files, and results from logs, and notifies the user by mail of transfer completions. `uucico` acts as the "login shell" for UUCP login accounts. When the local `uucico` daemon calls a remote machine, it communicates directly with the remote `uucico` daemon during the session.

After all the required files have been created, `uucp`, `uuto`, and `uux` programs execute the `uucico` daemon to contact the remote computer. `uusched` and `Uutry` all execute `uucico`. See the `uucico(1M)` man page for details.

`uuxqt` — Executes remote execution requests. This daemon searches the spool directory for execute files (always named `X.`*file*) that have been sent from a remote computer. When an `X.`*file* file is found, `uuxqt` opens it to get the list of data files that are required for the execution. `uuxqt` then checks to see if the required data files are available and accessible. If the files are available, `uuxqt` checks the `Permissions` file to verify that it has permission to execute the requested command. The `uuxqt` daemon is executed by the `uudemon.hour` shell script, which is started by `cron`. See the `uuxqt(1M)` man page for details.

`uusched` — Schedules the queued work in the spool directory. `uusched` is initially run at boot time by the `uudemon.hour` shell script, which is started by `cron`. See the

uusched(1M) man page for details. Before starting the uucico daemon, uusched randomizes the order in which remote computers are called.

in.uucpd — Supports UUCP connections over networks. The inetd on the remote host invokes in.uucpd whenever a UUCP connection is established. uucpd then prompts for a login name. uucico on the calling host must respond with a login name. in.uucpd then prompts for a password, unless a password is not required. See the in.uucpd(1M) man page for details.

UUCP Administrative Programs

Most UUCP administrative programs are in /usr/lib/uucp. Most basic database files are in /etc/uucp. The only exception is uulog, which is in /usr/bin. The home directory of the uucp login ID is /usr/lib/uucp. When running the administrative programs through su or login, use the uucp user ID. The user ID owns the programs and spooled data files.

uulog — Displays the contents of a specified computer's log files. Log files are created for each remote computer with which your machine communicates. The log files record each use of uucp, uuto, and uux. See the uucp(1C) man page for details.

uucleanup — Cleans up the spool directory. uucleanup is normally executed from the uudemon.cleanup shell script, which is started by cron. See the uucleanup(1M) man page for details.

Uutry — Tests call-processing capabilities and does moderate debugging. Uutry invokes the uucico daemon to establish a communication link between your machine and the remote computer that you specify. See the Uutry(1M) man page for details.

uucheck — Checks for the presence of UUCP directories, programs, and support files. uucheck can also check certain parts of the /etc/uucp/Permissions file for obvious syntactic errors. See the uucheck(1M) man page for details.

UUCP User Programs

The UUCP user programs are in /usr/bin. You do not need special permission to use these programs.

cu — Connects your machine to a remote computer so that you can log in to both machines at the same time. cu enables you to transfer files or execute commands on either machine without dropping the initial link. See the cu(1C) man page for details.

`uucp`	Lets you copy a file from one machine to another machine. `uucp` creates work files and data files, queues the job for transfer, and calls the `uucico` daemon, which in turn attempts to contact the remote computer. See the `uucp`(1C) man page for details.
`uuto`	Copies files from the local machine to the public spool directory `/var/spool/uucppublic/receive` on the remote machine. Unlike `uucp`, which lets you copy a file to any accessible directory on the remote machine, `uuto` places the file in an appropriate spool directory and tells the remote user to pick the file up with `uupick`. See the `uuto`(1C) man page for details.
`uupick`	Retrieves files in `/var/spool/uucppublic/receive` when files are transferred to a computer by using `uuto`. See the `uuto`(1C) man page.
`uux`	Creates the work, data, and execute files that are needed to execute commands on a remote machine. See the `uux`(1C) man page for details.
`uustat`	Displays the status of requested transfers (`uucp`, `uuto`, or `uux`). `uustat` also provides a means of controlling queued transfers. See the `uustat`(1C) man page for details.

UUCP Database Files

A major part of UUCP setup is the configuration of the files that compose the UUCP database. These files are in the `/etc/uucp` directory. You need to edit these files to set up UUCP or `asppp` on your machine. The files include the following:

`Config`	Contains a list of variable parameters. You can manually set these parameters to configure the network.
`Devconfig`	Used to configure network communications.
`Devices`	Used to configure network communications.
`Dialcodes`	Contains dial-code abbreviations that can be used in the phone number field of `Systems` file entries. Though not required, `Dialcodes` can be used by `asppp` as well as UUCP.
`Dialers`	Contains character strings that are required to negotiate with modems to establish connections with remote computers. `Dialers` is used by `asppp` as well as UUCP.
`Grades`	Defines job grades, and the permissions that are associated with each job grade, which users can specify to queue jobs to a remote computer.
`Limits`	Defines the maximum number of simultaneous `uucicos`, `uuxqts`, and `uuscheds` that are permitted on your machine.

`Permissions`	Defines the level of access that is granted to remote hosts that attempt to transfer files or execute commands on your machine.
`Poll`	Defines machines that are to be polled by your system and when they are polled.
`Sysfiles`	Assigns different or multiple files to be used by `uucico` and `cu` as `Systems`, `Devices`, and `Dialers` files.
`Sysname`	Enables you to define a unique UUCP name for a machine, in addition to its TCP/IP host name.
`Systems`	Contains information that is needed by the `uucico` daemon, `cu`, and `asppp` to establish a link to a remote computer. This information includes the following: ■ Name of the remote host ■ Name of the connecting device associated with the remote host ■ Time when the host can be reached ■ Telephone number ■ Login ID ■ Password

Several other files can be considered part of the supporting database but are not directly involved in establishing a link and transferring files.

Configuring UUCP Database Files

The UUCP database consists of the files that are shown in "UUCP Database Files" on page 158. However, basic UUCP configuration involves only the following critical files:

- `/etc/uucp/Systems`
- `/etc/uucp/Devices`
- `/etc/uucp/Dialers`

Because `asppp` uses some of the UUCP databases, you should understand at minimum these critical database files if you plan to configure `asppp`. After these databases are configured, UUCP administration is fairly straightforward. Typically, you edit the `Systems` file first, then edit the `Devices` file. You can usually use the default `/etc/uucp/Dialers` file, unless you plan to add dialers that are not in the default file. In addition, you might also want to use the following files for basic UUCP and `asppp` configuration:

- `/etc/uucp/Sysfiles`
- `/etc/uucp/Dialcodes`
- `/etc/uucp/Sysname`

Because these files work closely with each other, you should understand all their contents before you make any changes. A change to an entry in one file might require a change to a related entry in another file. The remaining files that are listed in "UUCP Database Files" on page 158 are not as critically intertwined.

Note – `asppp` uses only the files that are described in this section. `asppp` does not use the other UUCP database files.

◆ ◆ ◆ CHAPTER 11

Administering UUCP (Tasks)

This chapter explains how to start UUCP operations after you have modified the database file that is relevant to your machines. The chapter contains procedures and troubleshooting information for setting up and maintaining UUCP on machines that run the Oracle Solaris OS, such as the following:

UUCP Administration (Task Map)

The following table provides pointers to the procedures that are covered in this chapter, in addition to a short description of each procedure.

TABLE 11–1 Task Map for UUCP Administration

Task	Description	For Instructions
Allow remote machines to have access to your system	Edit the `/etc/passwd` file to add entries to identify the machines that are permitted to access your system.	"How to Add UUCP Logins" on page 162
Start UUCP	Use the supplied shell scripts to start UUCP.	"How to Start UUCP" on page 163
Enable UUCP to work with TCP/IP	Edit `/etc/inetd.conf` and `/etc/uucp/Systems` files to activate UUCP for TCP/IP.	"How to Activate UUCP for TCP/IP" on page 165
Troubleshoot some common UUCP problems	Use diagnostic steps to check for faulty modems or ACUs.	"How to Check for Faulty Modems or ACUs" on page 167

TABLE 11–1 Task Map for UUCP Administration *(Continued)*

Task	Description	For Instructions
	Use diagnostic steps to debug transmissions.	"How to Debug Transmissions" on page 168

Adding UUCP Logins

For incoming UUCP (`uucico`) requests from remote machines to be handled properly, each machine has to have a login on your system.

▼ How to Add UUCP Logins

To allow a remote machine to access your system, you need to add an entry to the `/etc/passwd` file as follows:

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Edit the `/etc/passwd` file and add the entry to identify the machine that is permitted to access your system.**

A typical entry that you might put into the `/etc/passwd` file for a remote machine that is permitted to access your system with a UUCP connection would be as follows:

```
Ugobi:*:5:5:gobi:/var/spool/uucppublic:/usr/lib/uucp/uucico
```

By convention, the login name of a remote machine is the machine name preceded by the uppercase letter `U`. Note that the name should not exceed eight characters. Otherwise, you might have to truncate or abbreviate the name.

The previous entry shows that a login request by `Ugobi` is answered by `/usr/lib/uucp/uucico`. The home directory is `/var/spool/uucppublic`. The password is obtained from the `/etc/shadow` file. You must coordinate the password and the login name with the UUCP administrator of the remote machine. The remote administrator must then add an appropriate entry, with login name and unencrypted password, in the remote machine's `Systems` file.

3 **Coordinate your machine name with the UUCP administrators on other systems.**

Similarly, you must coordinate your machine's name and password with the UUCP administrators of all machines that you want to reach through UUCP.

Starting UUCP

UUCP includes four shell scripts that poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. The scripts are as follows:

- `uudemon.poll`
- `uudemon.hour`
- `uudemon.admin`
- `uudemon.cleanup`

These shell scripts should execute regularly to ensure that UUCP runs smoothly. The `crontab` file to run the scripts is automatically created in `/usr/lib/uucp/uudemon.crontab` as part of the Oracle Solaris installation process, if you select the full installation. Otherwise, the file is created when you install the UUCP package.

You can also run the UUCP shell scripts manually. The following is the prototype `uudemon.crontab` file that you can tailor for a particular machine:

```
#
#ident  "@(#)uudemon.crontab    1.5     97/12/09 SMI"
#
# This crontab is provided as a sample. For systems
# running UUCP edit the time schedule to suit, uncomment
# the following lines, and use crontab(1) to activate the
# new schedule.
#
#48 8,12,16 * * * /usr/lib/uucp/uudemon.admin
#20 3 * * * /usr/lib/uucp/uudemon.cleanup
#0 * * * * /usr/lib/uucp/uudemon.poll
#11,41 * * * * /usr/lib/uucp/uudemon.hour
```

Note – By default, UUCP operations are disabled. To enable UUCP, edit the time schedule and uncomment the appropriate lines in the `uudemon.crontab` file.

▼ How to Start UUCP

To activate the `uudemon.crontab` file, do the following:

1. **Become an administrator.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Edit the `/usr/lib/uucp/uudemon.crontab` file and change entries as required.**

3. **Activate the `uudemon.crontab` file by issuing the following command:**

   ```
   crontab < /usr/lib/uucp/uudemon.crontab
   ```

uudemon.poll Shell Script

The default uudemon.poll shell script reads the /etc/uucp/Poll file once an hour. If any machines in the Poll file are scheduled to be polled, a work file (C.*sysnxxxx*) is placed in the /var/spool/uucp/*nodename* directory. *nodename* represents the UUCP node name of the machine.

The shell script is scheduled to run once an hour, before uudemon.hour, so that the work files are in place when uudemon.hour is called.

uudemon.hour Shell Script

The default uudemon.hour shell script does the following:

- Calls the uusched program to search the spool directories for work files (C.) that have not been processed. The script then schedules these files for transfer to a remote machine.
- Calls the uuxqt daemon to search the spool directories for execute files (X.) that have been transferred to your computer and were not processed when they were transferred.

By default, uudemon.hour runs twice an hour. You might want uudemon.hour to run more often if you expect high failure rates of calls to remote machines.

uudemon.admin Shell Script

The default uudemon.admin shell script does the following:

- Runs the uustat command with p and q options. The q reports on the status of work files (C.), data files (D.), and execute files (X.) that are queued. The p prints process information for networking processes that are listed in the lock files (/var/spool/locks).
- Sends resulting status information to the uucp administrative login by using mail.

uudemon.cleanup Shell Script

The default uudemon.cleanup shell script does the following:

- Collects log files for individual machines from the /var/uucp/.Log directory, merges these files, and places the files in the /var/uucp/.Old directory with other old log information
- Removes work files (C.) seven days old or older, data files (D.) seven days old or older, and execute files (X.) two days old or older from the spool files
- Returns mail that cannot be delivered to the sender
- Mails a summary of the status information that was gathered during the current day to the UUCP administrative login (uucp)

Running UUCP Over TCP/IP

To run UUCP on a TCP/IP network, you need to make a few modifications, as described in this section.

▼ How to Activate UUCP for TCP/IP

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Edit the `/etc/uucp/Systems` file to ensure that the entries have the following fields:**

System-Name Time TCP Port networkname Standard-Login-Chat

A typical entry would resemble the following:

```
rochester Any TCP - ur-seneca login: Umachine password: xxx
```

Notice that the *networkname* field permits you to specify explicitly the TCP/IP host name. This capability is important for some sites. In the previous example, the site has the UUCP node name `rochester`, which is different from its TCP/IP host name `ur-seneca`. Moreover, a completely different machine could easily run UUCP and have the TCP/IP host name of `rochester`.

The Port field in the `Systems` file should have the entry `-`. This syntax is equivalent to listing the entry as `uucp`. In almost every situation, the *networkname* is the same as the system name, and the Port field is `-`, which says to use the standard uucp port from the `services` database. The `in.uucpd` daemon expects the remote machine to send its login and password for authentication, and `in.uucpd` prompts for them, much as `getty` and `login` do.

3 **Edit the `/etc/inet/services` file to set up a port for UUCP:**

```
uucp    540/tcp    uucpd        # uucp daemon
```

You should not have to change the entry. However, if your machine runs NIS as its name service, you should make sure that the `config/service` of the `svc:/system/name-service/switch` service checks for `files` before `nis`. If the `config/service` property is not defined, then check the `config/default` property.

4 **Verify that UUCP is enabled.**

```
# svcs network/uucp
```

The UUCP service is managed by the Service Management Facility. To query the status of this service, you can use the `svcs` command. For an overview of the Service Management Facility, refer to Chapter 2, "Managing Services (Overview)," in *Managing Services and Faults in Oracle Solaris 11.1*.

5 **(Optional) If necessary, enable UUCP by typing the following:**

```
# inetadm -e network/uucp
```

UUCP Security and Maintenance

After you have set up UUCP, maintenance is straightforward. This section explains ongoing UUCP tasks that relate to security, maintenance, and troubleshooting.

Setting Up UUCP Security

The default `/etc/uucp/Permissions` file provides the maximum amount of security for your UUCP links. The default `Permissions` file contains no entries.

You can set additional parameters for each remote machine to define the following:

- Ways that the remote machine can receive files from your machine
- Directories for which the remote machine has read and write permission
- Commands that the remote machine can use for remote execution

A typical `Permissions` entry follows:

```
MACHINE=datsun LOGNAME=Udatsun VALIDATE=datsun
COMMANDS=rmail REQUEST=yes SENDFILES=yes
```

This entry allows files to be sent and be received to and from the "normal" UUCP directories, not from anywhere in the system. The entry also causes the UUCP user name to be validated at login time.

Regular UUCP Maintenance

UUCP does not require much maintenance. However, you must ensure that the `crontab` file is in place, as described in the section "How to Start UUCP" on page 163. Your concern should be the growth of mail files and the public directory.

Email for UUCP

All email messages that are generated by the UUCP programs and scripts are sent to the user ID `uucp`. If you do not log in frequently as that user, you might not realize that mail is accumulating and consuming disk space. To solve this problem, create an alias in `/etc/mail/aliases` and redirect that email either to `root` or to yourself and others who are responsible for maintaining UUCP. Remember to run the `newaliases` command after modifying the `aliases` file.

UUCP Public Directory

The directory `/var/spool/uucppublic` is the one place in every system to which UUCP by default is able to copy files. Every user has permission to change to `/var/spool/uucppublic` and read and write files in the directory. However, the directory's sticky bit is set, so the directory's mode is 01777. As a result, users cannot remove files that have been copied to it and that belong to `uucp`. Only you, as UUCP administrator logged in as `root` or `uucp`, can remove files from this directory. To prevent the uncontrolled accumulation of files in this directory, you should ensure that you remove files from it periodically.

If this maintenance is inconvenient for users, encourage them to use `uuto` and `uupick` rather than removing the sticky bit, which is set for security reasons. See the `uuto`(1C) man page for instructions for using `uuto` and `uupick`. You can also restrict the mode of the directory to only one group of people. If you do not want to risk someone filling your disk, you can even deny UUCP access to it.

Troubleshooting UUCP

These procedures describe how to solve common UUCP problems.

▼ How to Check for Faulty Modems or ACUs

You can check if the modems or other ACUs are not working properly in several ways.

1. **Become an administrator.**

   For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2. **Obtain counts and reasons for contact failure by running the following command:**

   ```
   # uustat -q
   ```

3. **Call over a particular line and print debugging information on the attempt.**

   The line must be defined as `direct` in the `/etc/uucp/Devices` file. You must add a telephone number to the end of the command line if the line is connected to an autodialer or the device must be set up as `direct`. Type:

   ```
   # cu -d -lline
   ```

   line is `/dev/cua/a`.

How to Debug Transmissions

If you cannot contact a particular machine, you can check communications to that machine with `Uutry` and `uucp`.

1 Become an administrator.

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 Try to make contact:

```
# /usr/lib/uucp/Uutry -r machine
```

Replace *machine* with the host name of the machine you are unable to contact. This command does the following:

- Starts the transfer daemon (`uucico`) with debugging. You can get more debugging information if you are `root`.
- Directs the debugging output to `/tmp/`*machine*.
- Prints the debugging output to your terminal by issuing the following command:

  ```
  # tail -f
  ```

  Press Control-C to end output. You can copy the output from `/tmp/`*machine* if you want to save the output.

3 If `Uutry` does not isolate the problem, try to queue a job:

```
# uucp -r file machine\!/dir/file
```

file	Use the name of the file that you want to transfer.
machine	Use the name of the machine that you want to copy to.
/dir/file	Specify the location of the file for the other machine.

4 Issue the following command:

```
# Uutry
```

If you still cannot solve the problem, you might need to call your local support representative. Save the debugging output, which can help diagnose the problem.

Note – You might also decrease or increase the level of debugging that is provided by `Uutry` through the `-x` *n* option. *n* indicates the debug level. The default debug level for `Uutry` is 5.

Debug level 3 provides basic information about when and how the connection is established, but not much information about the transmission. Debug level 9, however, provides exhaustive information about the transmission process. Be aware that debugging occurs at both ends of the transmission. If you intend to use a level higher than 5 on a moderately large text, contact the other site's administrator and decide when to change the level.

Checking the UUCP `/etc/uucp/Systems` File

Verify that you have up-to-date information in your `Systems` file if you are having trouble contacting a particular machine. Some information that might be out of date for a machine is the following:

- Phone number
- Login ID
- Password

Checking UUCP Error Messages

UUCP has two types of error messages: `ASSERT` and `STATUS`.

- When a process is aborted, `ASSERT` error messages are recorded in `/var/uucp/.Admin/errors`. These messages include the file name, `sccsid`, line number, and text. These messages usually result from system problems.
- `STATUS` error messages are stored in the `/var/uucp/.Status` directory. The directory contains a separate file for each remote machine that your computer attempts to communicate with. These files contain status information about attempted communication and whether the communication was successful.

Checking Basic Information

Several commands are available for checking basic networking information:

- Use the `uuname` command to list those machines that your machine can contact.
- Use the `uulog` command to display the contents of the log directories for particular hosts.
- Use the `uucheck -v` command to check for the presence of files and directories that are needed by `uucp`. This command also checks the `Permissions` file and displays information about the permissions that you have set up.

◆ ◆ ◆ CHAPTER 12

UUCP (Reference)

This chapter provides reference information for working with UUCP. The following topics are covered:

UUCP `/etc/uucp/Systems` File

The `/etc/uucp/Systems` file contains the information that is needed by the `uucico` daemon to establish a communication link to a remote computer. `/etc/uucp/Systems` is the first file that you need to edit to configure UUCP.

Each entry in the `Systems` file represents a remote computer with which your host communicates. A particular host can have more than one entry. The additional entries represent alternative communication paths that are tried in sequential order. In addition, by default UUCP prevents any computer that does not appear in `/etc/uucp/Systems` from logging in to your host.

By using the `Sysfiles` file, you can define several files to be used as `Systems` files. See "UUCP `/etc/uucp/Sysfiles` File" on page 189 for a description of `Sysfiles`.

The following is the syntax for an entry in the `Systems` file:

```
System-Name    Time    Type    Speed    Phone    Chat Script
```

See the following example of an entry in the Systems file.

EXAMPLE 12–1 Entry in /etc/uucp/Systems

```
Arabian     Any  ACUEC 38400 111222  ogin: Puucp ssword:beledi
```

Arabian	Entry for the System-Name field. For more information, see "System-Name Field in /etc/uucp/Systems File" on page 172.
Any	Entry for the Time field. For more information, see "Time Field in /etc/uucp/Systems File" on page 172.
ACUEC	Entry for the Type field. For more information, see "Type Field in /etc/uucp/Systems File" on page 173.
38400	Entry for the Speed field. For more information, see "Speed Field in /etc/uucp/Systems File" on page 174.
111222	Entry for the Phone field. For more information, see "Phone Field in /etc/uucp/Systems File" on page 174.
ogin: Puucp ssword:beledi	Entry for the Chat Script field. For more information, see "Chat-Script Field in /etc/uucp/Systems File" on page 175.

System-Name Field in /etc/uucp/Systems File

This field contains the node name of the remote computer. On TCP/IP networks, this name can be the machine's host name or a name that is created specifically for UUCP communications through the /etc/uucp/Sysname file. See "UUCP /etc/uucp/Systems File" on page 171. In Example 12–1, the System-Name field contains an entry for remote host Arabian.

Time Field in /etc/uucp/Systems File

This field specifies the day of week and time of day when the remote computer can be called. The format of the Time field follows:

```
daytime[;retry]
```

day Portion of Time Field

The *day* portion can be a list that contains some of the following entries.

Su Mo Tu We Th Fr Sa	For individual days.

`Wk`	For any weekday.
`Any`	For any day.
`Never`	Your host never initiates a call to the remote computer. The call must be initiated by the remote computer. Your host is then operating in *passive mode*.

time Portion of Time Field

Example 12–1 shows `Any` in the Time field, which indicates that host `Arabian` can be called at any time.

The *time* portion should be a range of times that are specified in 24-hour notation, for example, `0800-1230` for 8:30 a.m. to 12:30 p.m. If no *time* portion is specified, any time of day is assumed to be allowed for the call.

A time range that spans 0000 is permitted. For example, `0800-0600` means all times are allowed other than times between 6 a.m. and 8 a.m.

retry Portion of Time Field

The *retry* subfield enables you to specify the minimum time (in minutes) before a retry, following a failed attempt. The default wait is 60 minutes. The subfield separator is a semicolon (;). For example, `Any;9` is interpreted as call any time, but wait at least 9 minutes before retrying after a failure occurs.

If you do not specify a *retry* entry, an exponential back-off algorithm is used. This means that UUCP starts with a default wait time that grows larger as the number of failed attempts increases. For example, suppose the initial retry time is 5 minutes. If no response occurs, the next retry is 10 minutes later. The next retry is 20 minutes later, and so on until the maximum retry time of 23 hours is reached. If *retry* is specified, the value specified is always the retry time. Otherwise, the back-off algorithm is used.

Type Field in `/etc/uucp/Systems` File

This field contains the device type that should be used to establish the communication link to the remote computer. The keyword that is used in this field is matched against the first field of `Devices` file entries.

EXAMPLE 12–2 Keyword With the Type Field

```
Arabian    Any    ACUEC, g    38400    1112222    ogin: Puucp ssword:beledi
```

You can define the protocol that is used to contact the system by adding the protocol to the Type field. The previous example shows how to attach the protocol g to the device type ACUEC. For information about protocols, see "Protocol Definitions in /etc/uucp/Devices File" on page 183.

Speed Field in /etc/uucp/Systems File

This field, also known as the Class field, specifies the transfer speed of the device that is used in establishing the communication link. The UUCP speed field can contain a letter and speed, such as C1200 or D1200, to differentiate between classes of dialers. Refer to "Class Field in the /etc/uucp/Devices File" on page 180.

Some devices can be used at any speed, so the keyword Any can be used. This field must match the Class field in the associated Devices file entry.

EXAMPLE 12–3 Entry in Speed Field

```
eagle    Any    ACU, g    D1200    NY3251    ogin: nuucp ssword:Oakgrass
```

If information is not required for this field, use a dash (-) as a placeholder for the field.

Phone Field in /etc/uucp/Systems File

This field enables you to specify the telephone number, known as a *token*, of the remote computer for automatic dialers, which are known as *port selectors*. The telephone number consists of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, the abbreviation must be listed in the Dialcodes file.

EXAMPLE 12–4 Entry in the Phone Field

```
nubian    Any    ACU       2400    NY555-1212    ogin: Puucp ssword:Passuan
eagle     Any    ACU, g    D1200   NY=3251       ogin: nuucp ssword:Oakgrass
```

In the Phone field, an equal sign (=) instructs the ACU to wait for a secondary dial tone before dialing the remaining digits. A dash (-) in the string instructs the ACU to pause four seconds before dialing the next digit.

If your computer is connected to a port selector, you can access other computers that are connected to that selector. The Systems file entries for these remote machines should not have a telephone number in the Phone field. Instead, this field should contain the token to be passed to the switch. In this way, the port selector knows the remote machine with which your host wants to communicate, usually just the system name. The associated Devices file entry should have a \D at the end of the entry to ensure that this field is not translated by using the Dialcodes file.

Chat-Script Field in /etc/uucp/Systems File

This field, also known as the Login field, contains a string of characters that is called a *chat-script*. The chat script contains the characters the local and remote machines must pass to each other in their initial conversation. Chat scripts have the following format:

expect send [expect send] ....

expect represents the string that the local host expects to receive from the remote host to initiate conversation. *send* is the string that the local host sends after the local host receives the *expect* string from the remote host. A chat script can have more than one expect-send sequence.

A basic chat script might contain the following:

- Login prompt that the local host expects to receive from the remote machine
- Login name that the local host sends to the remote machine in order to log in
- Password prompt that the local host expects to receive from the remote machine
- Password that the local host sends to the remote machine

The *expect* field can be composed of subfields of the following form:

expect[-send-expect]...

The *-send* is sent if the prior *expect* is not successfully read. The *-expect* that follows the *-send* is the next expected string.

For example, with strings `login--login`, the UUCP on the local host expects `login`. If UUCP receives `login` from the remote machine, UUCP goes to the next field. If UUCP does not receive `login`, UUCP sends a carriage return, then looks for `login` again. If the local computer initially does not expect any characters, use the characters "", for NULL string, in the *expect* field. All *send* fields are sent with a carriage return appended unless the *send* string is terminated with a `\c`.

The following is an example of a `Systems` file entry that uses an *expect-send* string:

```
sonora Any ACUEC 9600 2223333 "" \r \r ogin:-BREAK-ogin: Puucpx ssword:xyzzy
```

This example instructs UUCP on the local host to send two carriage returns and wait for `ogin:` (for `Login:`). If `ogin`: is not received, send a `BREAK`. When you do receive `ogin:`, send the login name `Puucpx`. When you receive `ssword:` (for `Password:`), send the password `xyzzy`.

The following table lists some useful escape characters.

TABLE 12–1 Escape Characters Used in the Chat-Script Field of the Systems File

Escape Character	Meaning
\b	Sends or expects a backspace character.
\c	If at the end of a string, suppresses the carriage return that is normally sent. Ignored otherwise.
\d	Delays 1–3 seconds before sending more characters.
\E	Starts echo checking. From this point forward, whenever a character is transmitted, UUCP waits for the character to be received before continuing its checks.
\e	Echoes check-off.
\H	Ignores one hangup. Use this option for dialback modems.
\K	Sends a BREAK character.
\M	Turns on CLOCAL flag.
\m	Turns off CLOCAL flag.
\n	Sends or expects a newline character.
\N	Sends a NULL character (ASCII NUL).
\p	Pauses for approximately 1/4 to 1/2 second.
\r	Sends or expects a carriage return.
\s	Sends or expects a space character.
\t	Sends or expects a tab character.
EOT	Sends an EOT, followed by newline twice.
BREAK	Sends a BREAK character.
ddd	Sends or expects the character that is represented by the octal digits (*ddd*).

Enabling Dialback Through the Chat Script

Some companies set up dial-in servers to handle calls from remote computers. For example, your company might have a dial-in server with a dialback modem that employees can call from their home computers. After the dial-in server identifies the remote machine, the dial-in server disconnects the link to the remote machine and then calls back the remote machine. The communications link is then reestablished.

You can facilitate dialback by using the \H option in the Systems file chat script at the place where dialback should occur. Include the \H as part of an expect string at the place where the dial-in server is expected to hang up.

For example, suppose the chat script that calls a dial-in server contains the following string:

```
INITIATED\Hogin:
```

The UUCP dialing facility on the local machine expects to receive the characters, INITIATED, from the dial-in server. After the characters, INITIATED, have been matched, the dialing facility flushes any subsequent characters that the dialing facility receives until the dial-in server hangs up. The local dialing facility then waits until it receives the next part of the expect string, the characters ogin:, from the dial-in server. When it receives the ogin:, the dialing facility then continues through the chat script.

A string of characters does not need to directly precede or follow the \H, as shown in the previous sample string.

Hardware Flow Control in /etc/uucp/Systems File

You can also use the pseudo-send STTY=*value* string to set modem characteristics. For instance, STTY=crtscts enables hardware flow control. STTY accepts all stty modes. See the stty(1) and termio(7I) man pages for complete details.

The following example enables hardware flow control in a Systems file entry:

```
unix Any ACU 2400 12015551212 "" \r ogin: Puucp ssword:Passuan "" \ STTY=crtscts
```

This pseudo-send string can also be used in entries in the Dialers file.

Setting Parity in /etc/uucp/Systems File

In some situations, you have to reset the parity because the system that you are calling checks port parity and drops the line if it is wrong. The expect-send couplet, "" P_ZERO, sets the high-order bit (parity bit) to 0. See this expect-send couplet in the following example:

```
unix Any ACU 2400 12015551212 "" P_ZERO "" \r ogin: Puucp ssword:Passuan
```

The following are parity couplets that can follow the expect-send couplet, "" P_ZERO:

"" P_EVEN Sets the parity to even, which is the default

"" P_ODD Sets the parity to odd

"" P_ONE Sets the parity bit to 1

These parity couplets can be inserted anywhere in the chat script. The parity couplets apply to all information in the chat script that follows "" P_ZERO, the expect-send couplet. A parity couplet can also be used in entries in the Dialers file. The following example includes the parity couplet, "" P_ONE:

```
unix Any ACU 2400 12015551212 "" P_ZERO "" P_ONE "" \r ogin: Puucp ssword:Passuan
```

UUCP /etc/uucp/Devices File

The /etc/uucp/Devices file contains information for all the devices that can be used to establish a link to a remote computer. These devices include ACUs (which include high-speed modems), direct links, and network connections.

An entry in the /etc/uucp/Devices file has the following syntax:

```
Type   Line   Line2   Class   Dialer-Token-Pairs
```

The following is an entry in the Devices file for a U.S. Robotics V.32bis modem that is attached to port A and is running at 38,400 bps.

```
ACUEC   cua/a   -   38400   usrv32bis-ec
```

ACUEC — Entry in the Type field. For more information, see "Type Field in /etc/uucp/Devices File" on page 178.

cua/a — Entry in the Line field. For more information, see "Line Field in the /etc/uucp/Devices File" on page 180.

\- — Entry in the Line2 field. For more information, see "Line2 Field in the /etc/uucp/Devices File" on page 180.

38400 — Entry in the Class field. For more information, see "Class Field in the /etc/uucp/Devices File" on page 180.

usrv32bis-ec — Entry in the Dialer-Token-Pairs field. For more information, see "Dialer-Token-Pairs Field in the /etc/uucp/Devices File" on page 181.

Each field is described in the next section.

Type Field in /etc/uucp/Devices File

This field describes the type of link that the device establishes. The UUCP Type field can contain one of the keywords that is described in the sections that follow.

Direct Keyword

The Direct keyword appears mainly in entries for cu connections. This keyword indicates that the link is a direct link to another computer or a port selector. Create a separate entry for each line that you want to reference through the -l option of cu.

ACU Keyword

The ACU keyword indicates that the link to a remote computer (whether through cu, UUCP, asppp, or Solaris PPP 4.0) is made through a modem. This modem can be connected either directly to your computer or indirectly through a port selector.

Port Selector

The port selector is a variable that is replaced in the Type field by the name of a port selector. Port selectors are devices that are attached to a network that prompts for the name of a calling modem, then grant access. The file /etc/uucp/Dialers contains caller scripts only for the micom and develcon port selectors. You can add your own port selector entries to the Dialers file. See "UUCP /etc/uucp/Dialers File" on page 184 for more information.

System-Name Variable

This variable is replaced by the name of a machine in the Type field, indicating that the link is a direct link to this particular computer. This naming scheme is used to associate the line in this Devices entry with an entry in /etc/uucp/Systems for the computer *System-Name*.

Type Fields in Devices File and Systems File

Example 12–5 shows a comparison of the fields in /etc/uucp/Devices and the fields in /etc/uucp/Systems. The keyword that is used in the Type field of the Devices file is matched against the third field of the Systems file entries. In the Devices file, the Type field has the entry ACUEC, indicating an automatic call unit, in this instance a V.32bis modem. This value is matched against the Type field in the Systems file, which also contains the entry ACUEC. See "UUCP /etc/uucp/Systems File" on page 171 for more information.

EXAMPLE 12–5 Comparison of Type Fields in Devices file and Systems File

The following is an example of an entry in the Devices file.

```
ACUEC cua/a - 38400 usrv32bis-ec
```

The following is an example of an entry in the Systems file.

```
Arabian Any ACUEC 38400 111222 ogin: Puucp ssword:beledi
```

Line Field in the /etc/uucp/Devices File

This field contains the device name of the line (known as port) that is associated with the Devices entry. If the modem that is associated with a particular entry were attached to the /dev/cua/a device (serial port A), the name that is entered in this field would be cua/a. An optional modem control flag, M, can be used in the Line field to indicate that the device should be opened without waiting for a carrier. For example:

```
cua/a,M
```

Line2 Field in the /etc/uucp/Devices File

This field is a placeholder. Always use a hyphen (-) here. 801–type dialers, which are not supported in the Oracle Solaris OS, use the Line2 field. Non-801 dialers do not normally use this configuration, but still require a hyphen in this field.

Class Field in the /etc/uucp/Devices File

The Class field contains the speed of the device, if the keyword ACU or Direct is used in the Type field. However, the Class field can contain a letter and a speed, such as C1200 or D1200, to differentiate between classes of dialers, such as Centrex or Dimension PBX.

This differentiation is necessary because many larger offices can have more than one type of telephone network. One network might be dedicated to serving only internal office communications while another network handles the external communications. In such a situation, you must distinguish which line or lines should be used for internal communications and which should be used for external communications.

The keyword that is used in the Class field of the Devices file is matched against the Speed field of the Systems file.

EXAMPLE 12–6 Class Field in the Devices file

```
ACU   cua/a   -   D2400  hayes
```

Some devices can be used at any speed, so the keyword Any can be used in the Class field. If Any is used, the line matches any speed that is requested in the Speed field of the Systems file. If this field is Any and the Systems file Speed field is Any, the speed defaults to 2400 bps.

Dialer-Token-Pairs Field in the /etc/uucp/Devices File

The Dialer-Token-Pairs (DTP) field contains the name of a dialer and the token to pass it. The DTP field has this syntax:

dialer token [dialer token]

The *dialer* portion can be the name of a modem, a port monitor, or it can be `direct` or `uudirect` for a direct-link device. You can have any number of dialer-token pairs. If the *dialer* portion is not present, it is taken from a related entry in the `Systems` file. The *token* portion can be supplied immediately after the dialer portion.

The last dialer-token pair might not be present, depending on the associated dialer. In most situations, the last pair contains only a *dialer* portion. The *token* portion is retrieved from the Phone field of the associated `Systems` file entry.

A valid entry in the *dialer* portion can be defined in the `Dialers` file or can be one of several special dialer types. These special dialer types are compiled into the software and are therefore available without having entries in the `Dialers` file. The following list shows the special dialer types.

`TCP`	TCP/IP network
`TLI`	Transport Level Interface Network (without STREAMS)
`TLIS`	Transport Level Interface Network (with STREAMS)

See "Protocol Definitions in /etc/uucp/Devices File" on page 183 for more information.

Structure of the Dialer-Token-Pairs Field in the /etc/uucp/Devices File

The DTP field can be structured four different ways, depending on the device that is associated with the entry.

See the first way that the DTP field can be structured:

Directly connected modem – If a modem is connected directly to a port on your computer, the DTP field of the associated `Devices` file entry has only one pair. This pair would normally be the name of the modem. This name is used to match the particular `Devices` file entry with an entry in the `Dialers` file. Therefore, the Dialer field must match the first field of a `Dialers` file entry.

EXAMPLE 12–7 Dialers Field for Directly Connect Modem

```
Dialers   hayes =,-,  ""          \\dA\pTE1V1X1Q0S2=255S12=255\r\c
                                   \EATDT\T\r\c CONNECT
```

Notice that only the dialer portion (`hayes`) is present in the DTP field of the `Devices` file entry. This means that the *token* to be passed on to the dialer (in this instance, the phone number) is taken from the Phone field of a `Systems` file entry. (`\T` is implied, as described in Example 12–9.)

See the second and third ways that the DTP field can be structured:

- **Direct link** – For a direct link to a particular computer, the DTP field of the associated entry would contain the keyword `direct`. This condition is true for both types of direct-link entries, `Direct` and *System-Name*. Refer to "Type Field in /etc/uucp/Devices File" on page 178.
- **Computers on the same port selector** – If a computer with which you intend to communicate is on the same port selector switch as your computer, your computer must first access the switch. The switch then makes the connection to the other computer. This type of entry has only one pair. The *dialer* portion is used to match a `Dialers` file entry.

EXAMPLE 12–8 UUCP Dialers Field for Computers on Same Port Selector

```
Dialers    develcon ,""   ""           \pr\ps\c est:\007 \E\D\e \007
```

As shown, the *token* portion is left blank. This designation indicates that it is retrieved from the `Systems` file. The `Systems` file entry for this computer contains the token in the Phone field, which is normally reserved for the phone number of the computer. Refer to "UUCP /etc/uucp/Systems File" on page 171 for details. This type of DTP contains an escape character (`\D`), which ensures that the content of the Phone field is not interpreted as a valid entry in the `Dialcodes` file.

See the fourth way that the DTP field can be structured:

Modems that are connected to port selector – If a high-speed modem is connected to a port selector, your computer must first access the port selector switch. The switch makes the connection to the modem. This type of entry requires two dialer-token-pairs. The *dialer* portion of each pair (the fifth and seventh fields of the entry) is used to match entries in the `Dialers` file, as follows.

EXAMPLE 12–9 UUCP Dialers Field for Modems Connected to Port Selector

```
develcon ""     ""    \pr\ps\c  est:\007    \E\D\e      \007
ventel   =&-%   t""   \r\p\r\c  $           <K\T%\r>\c  ONLINE!
```

In the first pair, `develcon` is the dialer and `vent` is the token that is passed to the Develcon switch to tell it which device, such as a Ventel modem, to connect to your computer. This token

is unique for each port selector, as each switch can be set up differently. After the Ventel modem has been connected, the second pair is accessed. Ventel is the dialer and the token is retrieved from the `Systems` file.

Two escape characters can appear in a DTP field:

- `\T` – Indicates that the Phone (*token*) field should be translated by using the `/etc/uucp/Dialcodes` file. This escape character is normally placed in the `/etc/uucp/Dialers` file for each caller script that is associated with a modem, such as Hayes, and U.S. Robotics. Therefore, the translation does not occur until the caller script is accessed.
- `\D` – Indicates that the Phone (*token*) field should not be translated by using the `/etc/uucp/Dialcodes` file. If no escape character is specified at the end of a `Devices` entry, the `\D` is assumed (default). A `\D` is also used in the `/etc/uucp/Dialers` file with entries that are associated with network switches `develcon` and `micom`.

Protocol Definitions in `/etc/uucp/Devices` File

You can define the protocol to use with each device in `/etc/uucp/Devices`. This specification is usually unnecessary because you can use the default or define the protocol with the particular system you are calling. Refer to "UUCP /etc/uucp/Systems File" on page 171 for details. If you do specify the protocol, you must use the following form:

Type`,`*Protocol* *[parameters]*

For example, you can use `TCP,te` to specify the TCP/IP protocol.

The following table shows the available protocols for the `Devices` file.

TABLE 12–2 Protocols Used in `/etc/uucp/Devices`

Protocol	Description
`t`	This protocol is commonly used for transmissions over TCP/IP and other reliable connections. `t` assumes error-free transmissions.
`g`	This protocol is UUCP's native protocol. `g` is slow, reliable, and good for transmission over noisy telephone lines.
`e`	This protocol assumes transmission over error-free channels that are message oriented, as opposed to byte-stream oriented, such as TCP/IP.
`f`	This protocol is used for transmission over X.25 connections. `f` relies on flow control of the data stream and is meant for working over links that can (almost) be guaranteed to be error free, specifically X.25/PAD links. A checksum is enacted over a whole file only. If a transport fails, the receiver can request retransmission or retransmissions.

Here is an example that shows a protocol designation for a device entry:

```
TCP,te - - Any TCP -
```

This example indicates that, for device TCP, you should try to use the t protocol. If the other end of the transmission refuses, use the e protocol.

Neither e nor t is appropriate for use over modems. Even if the modem assures error-free transmission, data can still be dropped between the modem and the CPU.

UUCP /etc/uucp/Dialers File

The /etc/uucp/Dialers file contains dialing instructions for commonly used modems. You probably do not need to change or add entries to this file unless you plan to use a nonstandard modem or plan to customize your UUCP environment. Nevertheless, you should understand what is in the file and how it relates to the Systems and Devices file.

The text specifies the initial conversation that must occur on a line before the line can be made available for transferring data. This conversation, known as a chat script, is usually a sequence of ASCII strings that is transmitted and is expected. A chat script is often used to dial a phone number.

As shown in the examples in "UUCP /etc/uucp/Devices File" on page 178, the fifth field in a Devices file entry is an index into the Dialers file or a special dialer type, such as TCP, TLI, or TLIS. The uucico daemon attempts to match the fifth field in the Devices file with the first field of each Dialers file entry. In addition, each odd-numbered Devices field, starting with the seventh position, is used as an index into the Dialers file. If the match succeeds, the Dialers entry is interpreted to perform the dialer conversation.

Each entry in the Dialers file has the following syntax:

```
dialer   substitutions   expect-send
```

The following example shows the entry for a U.S. Robotics V.32bis modem.

EXAMPLE 12–10 Entry in /etc/uucp/Dialers File

```
usrv32bis-e    =,-,  ""    dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&W\r\c OK\r
                           \EATDT\T\r\c CONNECT\s14400/ARQ STTY=crtscts
```

usrv32bis-e
: Entry in the Dialer field. The Dialer field matches the fifth and additional odd-numbered fields in the Devices file.

`=,-, ""`
: Entry in the Substitutions field. The Substitutions field is a translation string. The first of each pair of characters is mapped to the second character in the pair. This mapping is usually used to translate = and - into whatever the dialer requires for "wait for dial tone" and "pause."

`dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&W\r\c OK\r`
: Entry in Expect-Send field. The Expect-Send fields are character strings.

`\EATDT\T\r\c CONNECT\s14400/ARQ STTY=crtscts`
: More of the Expect-Send field.

The following example shows sample entries in the `Dialers` file, as distributed when you install UUCP as part of the Oracle Solaris installation program.

EXAMPLE 12–11 Excerpts From /etc/uucp/Dialers

```
penril    =W-P "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP > 9\c OK

ventel    =&-%    "" \r\p\r\c $ <K\T%%\r>\c ONLINE!

vadic    =K-K    "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE

develcon    ""    "" \pr\ps\c est:\007

\E\D\e \n\007 micom    ""    "" \s\c NAME? \D\r\c GO

hayes    =,-,    "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r \EATDT\T\r\c CONNECT

#   Telebit TrailBlazer
tb1200    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=2\r\c OK\r
\EATDT\T\r\c CONNECT\s1200
tb2400    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=3\r\c OK\r
\EATDT\T\r\c CONNECT\s2400
tbfast    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=255\r\c OK\r
\EATDT\T\r\c CONNECT\sFAST

# USrobotics, Codes, and DSI modems

dsi-ec  =,-,    "" \dA\pTE1V1X5Q0S2=255S12=255*E1*F3*M1*S1\r\c OK\r \EATDT\T\r\c
CONNECT\sEC STTY=crtscts,crtsxoff

dsi-nec =,-,    "" \dA\pTE1V1X5Q0S2=255S12=255*E0*F3*M1*S1\r\c OK\r \EATDT\T\r\c CONNECT
STTY=crtscts,crtsxoff

usrv32bis-ec =,-,  "" \dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&W\r\c OK\r \EATDT\T\r\c
CONNECT\s14400/ARQ STTY=crtscts,crtsxoff

usrv32-nec =,-, "" \dA\pT&FE1V1X1Q0S2=255S12=255&A0&H1&M0&B0&W\r\c OK\r \EATDT\T\r\c
CONNECT STTY=crtscts,crtsxoff

codex-fast =,-, "" \dA\pT&C1&D2*MF0*AA1&R1&S1*DE15*FL3S2=255S7=40S10=40*TT5&W\r\c OK\r
\EATDT\T\r\c CONNECT\s38400 STTY=crtscts,crtsxoff

tb9600-ec =W-,  "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=6\r\c OK\r
\EATDT\T\r\cCONNECT\s9600 STTY=crtscts,crtsxoff
```

EXAMPLE 12–11 Excerpts From /etc/uucp/Dialers *(Continued)*

```
tb9600-nec =W-, "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=6S180=0\r\c OK\r \EATDT\T\r\c
CONNECT\s9600 STTY=crtscts,crtsxoff
```

The following table lists escape characters that are commonly used in the send strings in the Dialers file.

TABLE 12–3 Backslash Characters for /etc/uucp/Dialers

Character	Description
\b	Sends or expects a backspace character.
\c	No newline or carriage return.
\d	Delays for approximately 2 seconds.
\D	Phone number or token without Dialcodes translation.
\e	Disables echo checking.
\E	Enables echo checking for slow devices.
\K	Inserts a Break character.
\n	Sends newline.
\nnn	Sends octal number. Additional escape characters that can be used are listed in the section "UUCP /etc/uucp/Systems File" on page 171.
\N	Sends or expects a NULL character (ASCII NUL).
\p	Pauses for approximately 12–14 seconds.
\r	Returns.
\s	Sends or expects a space character.
\T	Phone number or token with Dialcodes translation.

Here is a penril entry in the Dialers file:

```
penril =W-P "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP > 9\c OK
```

First, the substitution mechanism for the phone number argument is established so that any = is replaced with a W (wait for dial tone) and any - with a P (pause).

The handshake that is given by the remainder of the line works as listed:

- "" – Waits for nothing, which means proceed to the next step.
- \d – Delays 2 seconds, then sends a carriage return.

- > – Waits for a >.
- Q\c – Sends a Q without a carriage return.
- : – Expects a :.
- \d- – Delays 2 seconds, sends a - and a carriage return.
- > – Waits for a >.
- s\p9\c – Sends an s, pauses, sends a 9 with no carriage return.
-)-W\p\r\ds\p9\c-) – Waits for a). If) is not received, processes the string between the - characters as follows. Sends a W, pauses, sends a carriage return, delays, sends an s, pauses, sends a 9 without a carriage return, then waits for the).
- y\c – Sends a y with no carriage return.
- : – Waits for a :.
- \E\TP – \E enables echo checking. From this point forward, whenever a character is transmitted, UUCP waits for the character to be received before proceeding. Then, UUCP sends the phone number. The \T means to take the phone number that is passed as an argument. The \T applies the Dialcodes translation and the modem function translation that is specified by field 2 of this entry. Then \T sends a P and a carriage return.
- > – Waits for a >.
- 9\c – Sends a 9 without a newline.
- OK – Waits for the string OK.

Enabling Hardware Flow Control in the /etc/uucp/Dialers File

You can also use the pseudo-send STTY=*value* string to set modem characteristics. For instance, STTY=crtscts enables outbound hardware flow control. STTY=crtsxoff enables inbound hardware flow control. STTY=crtscts,crtsxoff enables both outbound and inbound hardware flow control.

STTY accepts all the stty modes. See the stty(1) and termio(7I) man pages.

The following example would enable hardware flow control in a Dialers entry:

```
dsi =,–, "" \dA\pTE1V1X5Q0S2=255S12=255*E1*F3*M1*S1\r\c OK\r \EATDT\T\r\c
CONNECT\sEC STTY=crtscts
```

This pseudo-send string can also be used in entries in the Systems file.

Setting Parity in the /etc/uucp/Dialers File

In some situations, you have to reset the parity because the system that you are calling checks port parity and drops the line if it is wrong. The expect-send couplet P_ZERO sets parity to zero:

```
foo =,-, "" P_ZERO "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r\EATDT\T\r\c CONNECT
```

The following are parity couplets that can follow the expect-send couplet:

"" P_EVEN Sets the parity to even, which is the default

"" P_ODD Sets the parity to odd

"" P_ONE Sets the parity to one

This pseudo-send string can also be used in entries in the Systems file.

Other Basic UUCP Configuration Files

You can use files in this section in addition to the Systems, Devices, and Dialers file when doing basic UUCP configuration.

UUCP /etc/uucp/Dialcodes File

The /etc/uucp/Dialcodes file enables you to define dial-code abbreviations that can be used in the Phone field in the /etc/uucp/Systems file. You can use the Dialcodes file to provide additional information about a basic phone number that is used by several systems at the same site.

Each entry has the following syntax:

```
Abbreviation   Dial-Sequence
```

Abbreviation This field provides the abbreviation that is used in the Phone field of the Systems file.

Dial-Sequence This field provides the dial sequence that is passed to the dialer when that particular Systems file entry is accessed.

Compare the fields in the two files. The following are the fields in the Dialcodes file.

```
Abbreviation   Dial-Sequence
```

The following are the fields in the Systems file.

```
System-Name   Time   Type   Speed   Phone   Chat Script
```

The following table contains sample content for the fields in a `Dialcodes` file.

TABLE 12–4 Entries in the `Dialcodes` File

Abbreviation	Dial-Sequence
`NY`	`1=212`
`jt`	`9+847`

In the first row, NY is the abbreviation to appear in the Phone field of the `Systems` file. For example, the `Systems` file might have the following entry:

```
NY5551212
```

When `uucico` reads `NY` in the `Systems` file, `uucico` searches the `Dialcodes` file for `NY` and obtains the dialing sequence `1=212`. `1=212` is the dialing sequence that is needed for any phone call to New York City. This sequence includes the number `1`, an "equal sign" (`=`) meaning pause and wait for a secondary dial tone, and the area code `212`. `uucico` sends this information to the dialer, then returns to the `Systems` file for the remainder of the phone number, `5551212`.

The entry `jt 9=847-` would work with a Phone field such as `jt7867` in the `Systems` file. When `uucico` reads the entry that contains `jt7867` in the `Systems` file, `uucico` sends the sequence `9=847-7867` to the dialer, if the token in the dialer-token pair is `\T`.

UUCP `/etc/uucp/Sysfiles` File

The `/etc/uucp/Sysfiles` file lets you assign different files to be used by `uucp` and `cu` as `Systems`, `Devices`, and `Dialers` files. For more information about `cu`, see the cu(1C) man page. You can use `Sysfiles` for the following:

- Different `Systems` files so that requests for login services can be made to different addresses than `uucp` services.
- Different `Dialers` files so that you can assign different handshaking for `cu` and `uucp`.
- Multiple `Systems`, `Dialers`, and `Devices` files. The `Systems` file in particular can become large, making the file more convenient to split into several smaller files.

The syntax of the `Sysfiles` file is as follows:

`service=`*w* `systems=`*x*`:`*x* `dialers=`*y*`:`*y* `devices=`*z*`:`*z*

w Represents `uucico`, `cu`, or both commands separated by a colon

x Represents one or more files to be used as the `Systems` file, with each file name separated by a colon and read in the order that it is presented

y Represents one or more files to be used as the `Dialers` file

z Represents one or more files to be used as the Devices file

Each file name is assumed to be relative to the /etc/uucp directory unless a full path is given.

The following sample, /etc/uucp/Sysfiles, defines a local Systems file (Local_Systems) in addition to the standard /etc/uucp/Systems file:

```
service=uucico:cu systems=Systems :Local_Systems
```

When this entry is in /etc/uucp/Sysfiles, both uucico and cu first check in the standard /etc/uucp/Systems. If the system being called does not have an entry in that file, or if the entries in the file fail, then both commands check /etc/uucp/Local_Systems.

As specified in the previous entry, cu and uucico share the Dialers and Devices files.

When different Systems files are defined for uucico and cu services, your machine stores two different lists of Systems. You can print the uucico list by using the uuname command or the cu list by using the uuname -C command. The following is another example of the file, which shows that the alternate files are consulted first and the default files are consulted if necessary:

```
service=uucico systems=Systems.cico:Systems
  dialers=Dialers.cico:Dialers \
devices=Devices.cico:Devices
  service=cu systems=Systems.cu:Systems \
dialers=Dialers.cu:Dialers \
  devices=Devices.cu:Devices
```

UUCP /etc/uucp/Sysname File

Every machine that uses UUCP must have an identifying name, often referred to as the *node name*. The node name appears in the remote machine's /etc/uucp/Systems file, along with the chat script and other identifying information. Normally, UUCP uses the same node name as is returned by the uname -n command, which is also used by TCP/IP.

You can specify a UUCP node name independent of the TCP/IP host name by creating the /etc/uucp/Sysname file. The file has a one-line entry that contains the UUCP node name for your system.

UUCP /etc/uucp/Permissions File

The /etc/uucp/Permissions file specifies the permissions that remote computers have for login, file access, and command execution. Some options restrict the remote computer's ability

to request files and its ability to receive files that are queued by the local machine. Another option is available that specifies the commands that a remote machine can execute on the local computer.

UUCP Structuring Entries

Each entry is a logical line, with physical lines terminated by a backslash (\) to indicate continuation. Entries are composed of options that are delimited by a blank space. Each option is a name-value pair in the following format:

name=value

Values can be colon-separated lists. No blank space is allowed within an option assignment.

Comment lines begin with a pound sign (#) and occupy the entire line up to a newline character. Blank lines are ignored, even within multiple-line entries.

The types of `Permissions` file entries are as follows:

- `LOGNAME` – Specifies the permissions that become effective when a remote computer logs in to (calls) your computer.

  Note – When a remote machine calls you, its identity is questionable unless the remote machine has a unique login and verifiable password.

- `MACHINE` – Specifies permissions that become effective when your computer logs in to (calls) a remote computer.

`LOGNAME` entries contain a `LOGNAME` option. `MACHINE` entries contain a `MACHINE` option. One entry can contain both options.

UUCP Considerations

When using the `Permissions` file to restrict the level of access that is granted to remote computers, you should consider the following:

- All login IDs that are used by remote computers to log in for UUCP communications must appear in one and only one `LOGNAME` entry.
- Any site that is called with a name that does not appear in a `MACHINE` entry has the following default permissions or restrictions:
  - Local send-and-receive requests are executed.

- The remote computer can send files to your computer's /var/spool/uucppublic directory.
- The commands that are sent by the remote computer for execution on your computer must be one of the default commands, usually rmail.

UUCP REQUEST Option

When a remote computer calls your computer and requests to receive a file, this request can be granted or be denied. The REQUEST option specifies whether the remote computer can request to set up file transfers from your computer. The string REQUEST=yes specifies that the remote computer can request to transfer files from your computer. The string REQUEST=no specifies that the remote computer cannot request to receive files from your computer. REQUEST=no, the default value, is used if the REQUEST option is not specified. The REQUEST option can appear in either a LOGNAME entry, so that the remote computer calls you, or a MACHINE entry, so that you call the remote computer.

UUCP SENDFILES Option

When a remote computer calls your computer and completes its work, the remote computer can attempt to retrieve the work that your computer has queued for it. The SENDFILES option specifies whether your computer can send the work that is queued for the remote computer.

The string SENDFILES=yes specifies that your computer can send the work that is queued for the remote computer if it is logged in as one of the names in the LOGNAME option. This string is *mandatory* if you have entered Never in the Time field of /etc/uucp/Systems. This designation sets up your local machine in passive mode, but it is not allowed to initiate a call to this particular remote computer. See "UUCP /etc/uucp/Systems File" on page 171 for more information.

The string SENDFILES=call specifies that files that are queued in your computer are sent only when your computer calls the remote computer. The call value is the default for the SENDFILES option. This option is only significant in LOGNAME entries because MACHINE entries apply when calls are sent to remote computers. If the option is used with a MACHINE entry, the option is ignored.

UUCP MYNAME Option

This option enables you to designate a unique UUCP node name for your computer in addition to its TCP/IP host name, as returned by the hostname command. For instance, if you have unknowingly given your host the same name as that of some other system, you can set the

MYNAME option of the Permissions file. Suppose that you want your organization to be known as widget. If all your modems are connected to a machine with the host name gadget, you can have an entry in gadget's Permissions file that reads as follows:

```
service=uucico systems=Systems.cico:Systems
  dialers=Dialers.cico:Dialers \
  devices=Devices.cico:Devices
service=cu systems=Systems.cu:Systems \
  dialers=Dialers.cu:Dialers \
  devices=Devices.cu:Devices
```

Now, the system world can log in to the machine gadget as if it were logging in to widget. In order for machine world to know you also by the aliased name widget when you call it, you can have an entry that reads as follows:

```
MACHINE=world MYNAME=widget
```

You can also use the MYNAME option for testing purposes, as this option allows your machine to call itself. However, because this option could be used to mask the real identity of a machine, you should use the VALIDATE option, as described in "UUCP VALIDATE Option" on page 196.

UUCP READ and WRITE Options

These options specify the various parts of the file system that uucico can read from or write to. You can designate READ and WRITE options with either MACHINE or LOGNAME entries.

The default for both the READ and WRITE options is the uucppublic directory, as shown in the following strings:

```
READ=/var/spool/uucppublic WRITE=/var/spool/uucppublic
```

The strings READ=/ and WRITE=/ specify permission to access any file that can be accessed by a local user with Other permissions.

The value of these entries is a colon-separated list of path names. The READ option is for requesting files, and the WRITE option is for depositing files. One of the values must be the prefix of any full path name of a file entering or exiting. To grant permission to deposit files in /usr/news as well as in the public directory, use the following values with the WRITE option:

```
WRITE=/var/spool/uucppublic:/usr/news
```

If the READ and WRITE options are used, all path names must be specified because the path names are not added to the default list. For instance, if the /usr/news path name were the only path specified in a WRITE option, permission to deposit files in the public directory would be denied.

Be careful which directories you make accessible for reading and writing by remote systems. For example, the /etc directory contains many critical system files. Remote users should not have permission to deposit files in this directory.

UUCP NOREAD and NOWRITE Options

The NOREAD and NOWRITE options specify exceptions to the READ and WRITE options or defaults. The following entry permits reading any file except those files in the /etc directory (and its subdirectories) Remember, these options are prefixes.

```
READ=/ NOREAD=/etc WRITE=/var/spool/uucppublic
```

This entry permits writing only to the default /var/spool/uucppublic directory. NOWRITE works in the same manner as the NOREAD option. You can use the NOREAD and NOWRITE options in both LOGNAME and MACHINE entries.

UUCP CALLBACK Option

You can use the CALLBACK option in LOGNAME entries to specify that no transaction occurs until the calling system is called back. The reasons to set up CALLBACK are as follows:

- For security purposes – If you call back a machine, you can be sure it is the right machine.
- For accounting purposes – If you are doing long data transmissions, you can choose the machine that is billed for the longer call.

The string CALLBACK=yes specifies that your computer must call back the remote computer before any file transfers can occur.

The default for the CALLBACK option is CALLBACK=no. If you set CALLBACK to yes, the permissions that affect the rest of the conversation must be specified in the MACHINE entry that corresponds to the caller. Do not specify these permissions in the LOGNAME, or in the LOGNAME entry that the remote machine might have set for your host.

Note – If two sites have the CALLBACK option set for each other, a conversation never is started.

UUCP COMMANDS Option

Caution – The COMMANDS option can compromise the security of your system. Use this option with extreme care.

You can use the COMMANDS option in MACHINE entries to specify the commands that a remote computer can execute on your machine. The uux program generates remote execution requests and queues the requests to be transferred to the remote computer. Files and commands are sent to the target computer for remote execution, which is an exception to the rule that MACHINE entries apply only when your system calls out.

Note that COMMANDS is not used in a LOGNAME entry. COMMANDS in MACHINE entries defines command permissions, whether you call the remote system or the remote system calls you.

The string COMMANDS=rmail specifies the default commands that a remote computer can execute on your computer. If a command string is used in a MACHINE entry, the default commands are overridden. For instance, the following entry overrides the COMMAND default so that the computers that are named owl, raven, hawk, and dove can now execute rmail, rnews, and lp on your computer.

```
MACHINE=owl:raven:hawk:dove COMMANDS=rmail:rnews:lp
```

In addition to the names as just specified, you can have full path names of commands. For example, the following entry specifies that command rmail uses the default search path.

```
COMMANDS=rmail:/usr/local/rnews:/usr/local/lp
```

The default search path for UUCP is /bin and /usr/bin. When the remote computer specifies rnews or /usr/local/rnews for the command to be executed, /usr/local/rnews is executed regardless of the default path. Likewise, /usr/local/lp is the lp command that is executed.

Including the ALL value in the list means that any command from the remote computers that are specified in the entry is executed. If you use this value, you give the remote computers full access to your machine.

Caution – This value allows far more access than normal users have. You should use this value only when both machines are at the same site, are closely connected, and the users are trusted.

Here is the string with the ALL value added:

```
COMMANDS=/usr/local/rnews:ALL:/usr/local/lp
```

This string illustrates two points:

- The ALL value can appear anywhere in the string.
- The path names that are specified for rnews and lp are used (instead of the default) if the requested command does not contain the full path names for rnews or lp.

You should use the VALIDATE option whenever you specify potentially dangerous commands, such as cat and uucp with the COMMANDS option. Any command that reads or writes files is potentially dangerous to local security when the command is executed by the UUCP remote execution daemon (uuxqt).

UUCP VALIDATE Option

Use the VALIDATE option in conjunction with the COMMANDS option whenever you specify commands that are potentially dangerous to your machine's security. VALIDATE is merely an added level of security on top of the COMMANDS option, though it is a more secure way to open command access than ALL.

VALIDATE provides a certain degree of verification of the caller's identity by cross-checking the host name of a calling machine against the login name it uses. The following string ensures that if any machine other than widget or gadget tries to log in as Uwidget, the connection is refused.

```
LOGNAME=Uwidget VALIDATE=widget:gadget
```

The VALIDATE option requires privileged computers to have a unique login and password for UUCP transactions. An important aspect of this validation is that the login and password that are associated with this entry are protected. If an outsider obtains that information, that particular VALIDATE option can no longer be considered secure.

Carefully consider which remote computers you are granting privileged logins and passwords for UUCP transactions. Giving a remote computer a special login and password with file access and remote execution capability is like giving anyone on that computer a normal login and password on your computer. Therefore, if you cannot trust someone on the remote computer, do not provide that computer with a privileged login and password.

The following LOGNAME entry specifies that if one of the remote computers that claims to be eagle, owl, or hawk logs in on your computer, it must have used the login uucpfriend:

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

If an outsider obtains the uucpfriend login and password, masquerading is easy.

But what does this entry have to do with the COMMANDS option, which appears only in MACHINE entries? This entry links the MACHINE entry (and COMMANDS option) with a LOGNAME entry that is associated with a privileged login. This link is needed because the execution daemon is not running while the remote computer is logged in. Actually, the link is an asynchronous process that does not know which computer sent the execution request. Therefore, the real question is: How does your computer know where the execution files came from?

Each remote computer has its own spool directory on your local machine. These spool directories have write permission that is given only to the UUCP programs. The execution files

from the remote computer are put in its spool directory after being transferred to your computer. When the uuxqt daemon runs, it can use the spool directory name to find the MACHINE entry in the Permissions file and get the COMMANDS list. Or, if the computer name does not appear in the Permissions file, the default list is used.

This example shows the relationship between the MACHINE and LOGNAME entries:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
COMMANDS=rmail:/usr/local/rnews \
READ=/ WRITE=/
LOGNAME=uucpz VALIDATE=eagle:owl:hawk \
REQUEST=yes SENDFILES=yes \
READ=/ WRITE=/
```

The value in the COMMANDS option means that remote users can execute rmail and /usr/local/rnews.

In the first entry, you must assume that when you want to call one of the computers that is listed, you are really calling either eagle, owl, or hawk. Therefore, any files that are put into one of the eagle, owl, or hawk spool directories is put there by one of those computers. If a remote computer logs in and says that it is one of these three computers, its execution files are also put in the privileged spool directory. You therefore have to validate that the computer has the privileged login uucpz.

UUCP MACHINE Entry for OTHER

You might want to specify different option values for remote machines that are not mentioned in specific MACHINE entries. The need might arise when many computers are calling your host, and the command set changes from time to time. The name OTHER for the computer name is used for this entry as shown in this example:

```
MACHINE=OTHER \
COMMANDS=rmail:rnews:/usr/local/Photo:/usr/local/xp
```

All other options that are available for the MACHINE entry can also be set for the computers that are not mentioned in other MACHINE entries.

Combining MACHINE and LOGNAME Entries for UUCP

You can combine MACHINE and LOGNAME entries into a single entry when the common options are the same. For example, the two sets of entries that follow share the same REQUEST, READ, and WRITE options:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
READ=/ WRITE=/
```

and

```
LOGNAME=uupz REQUEST=yes SENDFILES=yes \
READ=/ WRITE=/
```

You can merge these entries, as shown:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
logname=uucpz SENDFILES-yes \
READ=/ WRITE=/
```

Combining MACHINE and LOGNAME entries makes the Permissions file more manageable and efficient.

UUCP Forwarding

When sending files through a series of machines, the intermediary machines must have the command uucp among their COMMANDS options. If you type the following command, the forwarding operation works only if machine willow permits machine oak to execute the uucp program.

```
% uucp sample.txt oak\!willow\!pine\!/usr/spool/uucppublic
```

The machine oak also must permit your machine to execute the uucp program. The machine pine, as the last machine designated, does not have to permit the uucp command because the machine is not doing any forwarding operations. Machines are not normally set up this way.

UUCP /etc/uucp/Poll File

The /etc/uucp/Poll file contains information for polling remote computers. Each entry in the Poll file contains the name of a remote computer to call, followed by a tab character or a space, and finally the hours the computer should be called. The format of entries in the Poll file are as follows:

sys-name hour ...

For example, the entry **eagle 0 4 8 12 16 20** provides polling of computer eagle every four hours.

The uudemon.poll script processes the Poll file but does not actually perform the poll. The script merely sets up a polling work file (always named C.*file*) in the spool directory. The uudemon.poll script starts the scheduler, and the scheduler examines all work files in the spool directory.

UUCP /etc/uucp/Config File

The /etc/uucp/Config file enables you to override certain parameters manually. Each entry in the Config file has this format:

parameter=value

See the Config file that is provided with your system for a complete list of configurable parameter names.

The following Config entry sets the default protocol ordering to Gge and changes the G protocol defaults to 7 windows and 512-byte packets.

```
Protocol=G(7,512)ge
```

UUCP/etc/uucp/Grades File

The /etc/uucp/Grades file contains the definitions for the job grades that can be used to queue jobs to a remote computer. This file also contains the permissions for each job grade. Each entry in this file represents a definition of an administrator-defined job grade that lets users queue jobs.

Each entry in the Grades file has the following format:

User-job-grade System-job-grade Job-size Permit-type ID-list

Each entry contains fields that are separated by a blank space. The last field in the entry is composed of subfields that are also separated by spaces. If an entry occupies more than one physical line, you can use a backslash to continue the entry onto the following line. Comment lines begin with a pound sign (#) and occupy the entire line. Blank lines are always ignored.

UUCP User-job-grade Field

This field contains an administrator-defined user-job-grade name of up to 64 characters.

UUCP System-job-grade Field

This field contains a single-character job grade to which *User-job-grade* is mapped. The valid list of characters is A–Z, a–z, with A having the highest priority and z the lowest.

Relationship Between User and System Job Grades

The user job grade can be bound to more than one system job grade. Note that the `Grades` file is searched sequentially for occurrences of a user job grade. Therefore, any multiple occurrences of a system job grade should be listed in compliance with the restriction on the maximum job size.

While no maximum number exists for the user job grades, the maximum number of system job grades that are allowed is 52. The reason is that more than one *User-job-grade* can be mapped to a *System-job-grade*, but each *User-job-grade* must be on a separate line in the file. Here is an example:

```
mail N Any User Any netnews N Any User Any
```

If this configuration is in a `Grades` file, these two *User-job-grade* fields share the same *System-job-grade*. Because the permissions for a *Job-grade* are associated with a *User-job-grade* and not a *System-job-grade*, two *User-job-grade*s can share the same *System-job-grade*s and have two different sets of permissions.

Default Grade

You can define the binding of a default *User-job-grade* to a system job grade. You must use the keyword default as the user job grade in the *User-job-grade* field of the `Grades` file and the system job grade that it is bound to. The Restrictions and ID fields should be defined as `Any` so that any user and any size job can be queued to this grade. Here is an example:

```
default a Any User Any
```

If you do not define the default user job grade, the built-in default grade `Z` is used. Because the restriction field default is `Any`, multiple occurrences of the default grade are not checked.

UUCP Job-size Field

This field specifies the maximum job size that can be entered in the queue. *Job-size* is measured in bytes and can be a list of the options that are described in the following list.

nnnn	Integer that specifies the maximum job size for this job grade
n`K`	Decimal number that represents the number of kilobytes (`K` is an abbreviation for kilobyte)
n`M`	Decimal number that represents the number of megabytes (`M` is an abbreviation for megabyte)
`Any`	Keyword that specifies that no maximum job size exists

Here are some examples:

- `5000` represents 5000 bytes
- `10K` represents 10 Kbytes
- `2M` represents 2 Mbytes

UUCP Permit-type Field

This field contains a keyword that denotes how to interpret the ID list. The following table lists the keywords and their meanings.

TABLE 12–5 Permit-type Field

Keyword	ID List Contents
`User`	Login names of users who are permitted to use this job grade
`Non-user`	Login names of users who are not permitted to use this job grade
`Group`	Group names whose members are permitted to use this group
`Non-group`	Group names whose members are not permitted to use this job grade

UUCP ID-list Field

This field contains a list of login names or group names that are to be permitted or denied queuing to this job grade. The list of names are separated by a blank space and terminated by a newline character. The keyword `Any` is used to denote that anyone is permitted to queue to this job grade.

Other UUCP Configuration Files

This section describes three less-frequently modified files that impact the use of UUCP facilities.

UUCP `/etc/uucp/Devconfig` File

The `/etc/uucp/Devconfig` file enables you to configure devices by service, such as `uucp` or `cu`. `Devconfig` entries define the STREAMS modules that are used for a particular device. These entries have the following format:

`service=`*x* `device=`*y* `push=`*z*[:*z*...]

x can be cu, uucico, or both services separated by a colon. *y* is the name of a network and must match an entry in the Devices file. *z* is replaced by the names of STREAMS modules in the order that they are to be pushed onto the Stream. Different modules and devices can be defined for cu and uucp services.

The following entries are for a STARLAN network and would most commonly be used in the file:

```
service=cu       device=STARLAN     push=ntty:tirdwr
service=uucico   device=STARLAN     push=ntty:tirdwr
```

This example pushes ntty, then tirdwr.

UUCP /etc/uucp/Limits File

The /etc/uucp/Limits file controls the maximum number of simultaneous uucicos, uuxqts, and uuscheds that are running in the uucp networking. In most situations, the default values are acceptable and no changes are needed. If you want to change them, however, use any text editor.

The format of the Limits file is as follows:

service=*x* max=*y*:

x can be uucico, uuxqt or uusched, and *y* is the limit that is permitted for that service. The fields can be in any order and in lowercase.

The following entries should most commonly be used in the Limits file:

```
service=uucico max=5
service=uuxqt max=5
service=uusched max=2
```

The example allows five uucicos, five uuxqts, and two uuscheds to run on your machine.

UUCP remote.unknown File

The other file that affects the use of communication facilities is the remote.unknown file. This file is a binary program that executes when a machine that is not found when any of the Systems files starts a conversation. This program logs the conversation attempt and drops the connection.

Caution – If you change the permissions of the remote.unknown file so that the file cannot execute, your system accepts connections from any system.

This program executes when a machine that is not in any of the `Systems` starts a conversation. The program logs the conversation attempt but fails to make a connection. If you change the permissions of this file so that the file cannot execute (`chmod 000 remote.unknown`), your system accepts any conversation requests. This change is not trivial. You should have good reasons for making this change.

UUCP Administrative Files

The UUCP administrative files are described next. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

- *Temporary data files* (`TM`) – These data files are created by UUCP processes under the spool directory `/var/spool/uucp/`*x* when a file is received from another computer. The directory *x* has the same name as the remote computer that is sending the file. The names of the temporary data files have the following format:

  `TM.`*pid.ddd*

  pid is a process ID and *ddd* is a sequential three-digit number that starts at 0.

  When the entire file is received, the `TM.`*pid.ddd* file is moved to the path name that is specified in the `C.`*sysnxxxx* file (discussed subsequently) that caused the transmission. If processing is abnormally terminated, the `TM.`*pid.ddd* file can remain in the *x* directory. These files should be automatically removed by `uucleanup`.
- *Lock files* (`LCK`) – Lock files are created in the `/var/spool/locks` directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The following table shows the different types of UUCP lock files.

TABLE 12–6 UUCP Lock Files

File Name	Description
`LCK.`*sys*	*sys* represents the name of the computer that is using the file
`LCK.`*dev*	*dev* represents the name of a device that is using the file
`LCK.`*LOG*	*LOG* represents a locked UUCP log file

These files can remain in the spool directory if the communications link is unexpectedly dropped, such as when a computer crashes. The lock file is ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock.

- *Work file* (`C.`) – Work files are created in a spool directory when work, such as file transfers or remote command executions, has been queued for a remote computer. The names of work files have the following format:

  `C.`*sysnxxxx*

sys is the name of the remote computer, *n* is the ASCII character that represents the grade (priority) of the work, and *xxxx* is the four-digit job sequence number that is assigned by UUCP. Work files contain the following information:

- Full path name of the file to be sent or be requested.
- Full path name of the destination or user or file name.
- User login name.
- List of options.
- Name of associated data files in the spool directory. If the `uucp -C` or `uuto -p` option was specified, a dummy name (`D.0`) is used.
- Mode bits of the source file.
- Remote user's login name to be notified on completion of the transfer.

- *Data file*(`D.`) – Data files are created when you specify on the command line to copy the source file to the spool directory. The names of data files have the following format:

  `D.`*systmxxxxyyy* – *systm* is the first five characters in the name of the remote computer. *xxxx* is a four-digit job sequence number assigned by `uucp`. The four-digit job sequence number can be followed by a subsequent number. *yyy* is used when several `D.` files are created for a work (`C.`) file.

- `X.` *(execute file)* – Execute files are created in the spool directory prior to remote command executions. The names of execute files have the following format:

  `X.`*sysnxxxx*

  sys is the name of the remote computer. *n* is the character that represents the grade (priority) of the work. *xxxx* is a four-digit sequence number that is assigned by UUCP. Execute files contain the following information:

  - Requester's login and computer name
  - Names of files that are required for execution
  - Input to be used as the standard input to the command string
  - Computer and file name to receive standard output from the command execution
  - Command string
  - Option lines for return status requests

UUCP Error Messages

This section lists the error messages that are associated with UUCP.

UUCP ASSERT Error Messages

The following table lists ASSERT error messages.

TABLE 12–7 ASSERT Error Messages

Error Message	Description or Action
CAN'T OPEN	An `open()` or `fopen()` failed.
CAN'T WRITE	A `write()`, `fwrite()`, `fprint()`, or similar command, failed.
CAN'T READ	A `read()`, `fgets()`, or similar command failed.
CAN'T CREATE	A `creat()` call failed.
CAN'T ALLOCATE	A dynamic allocation failed.
CAN'T LOCK	An attempt to make a `LCK` (lock) file failed. In some situations, this error is fatal.
CAN'T STAT	A `stat()` call failed.
CAN'T CHMOD	A `chmod()` call failed.
CAN'T LINK	A `link()` call failed.
CAN'T CHDIR	A `chdir()` call failed.
CAN'T UNLINK	An `unlink()` call failed.
WRONG ROLE	This is an internal logic problem.
CAN'T MOVE TO CORRUPTDIR	An attempt to move some bad `C.` or `X.` files to the `/var/spool/uucp/.Corrupt` directory failed. The directory is probably missing or has wrong modes or owner.
CAN'T CLOSE	A `close()` or `fclose()` call failed.
FILE EXISTS	The creation of a `C.` or `D.` file is attempted, but the file exists. This error occurs when a problem arises with the sequence file access, which usually indicates a software error.
NO uucp SERVICE NUMBER	A TCP/IP call is attempted, but no entry is in `/etc/services` for UUCP.
BAD UID	The user ID is not in the password database. Check name service configuration.
BAD LOGIN_UID	Same as previous description.
BAD LINE	A bad line is in the `Devices` file. Not enough arguments on one or more lines.
SYSLST OVERFLOW	An internal table in `gename.c` overflowed. A single job attempted to talk to more than 30 systems.
TOO MANY SAVED C FILES	Same as previous description.
RETURN FROM fixline ioctl	An `ioctl(2)`, which should never fail, failed. A system driver problem has occurred.
BAD SPEED	A bad line speed appears in the `Devices` or `Systems` file (Class or Speed field).
BAD OPTION	A bad line or option is in the `Permissions` file. This error must be fixed immediately.
PKCGET READ	The remote machine probably hung up. No action is needed.
PKXSTART	The remote machine aborted in a nonrecoverable way. This error can usually be ignored.

TABLE 12–7 ASSERT Error Messages *(Continued)*

Error Message	Description or Action
TOO MANY LOCKS	An internal problem has occurred. Contact your system vendor.
XMV ERROR	A problem with some file or directory has occurred. The spool directory is the probable cause, as the modes of the destinations were supposed to be checked before this process was attempted.
CAN'T FORK	An attempt to make a `fork` and `exec` failed. The current job should not be lost but will be attempted later (`uuxqt`). No action is needed.

UUCP STATUS Error Messages

The following table is a list of the most common STATUS error messages.

TABLE 12–8 UUCP STATUS Messages

Error Message	Description/Action
OK	Status is acceptable.
NO DEVICES AVAILABLE	Currently no device is available for the call. Check whether a valid device is in the `Devices` file for the particular system. Check the `Systems` file for the device to be used to call the system.
WRONG TIME TO CALL	A call was placed to the system at a time other than what is specified in the `Systems` file.
TALKING	Self-explanatory.
LOGIN FAILED	The login for the particular machine failed. The cause could be a wrong login or password, wrong number, a slow machine, or failure in executing the Dialer-Token-Pairs script.
CONVERSATION FAILED	The conversation failed after successful startup. This error usually means that one side went down, the program aborted, or the line (link) was dropped.
DIAL FAILED	The remote machine never answered. The cause could be a bad dialer or the wrong phone number.
BAD LOGIN/MACHINE COMBINATION	The machine called with a login/machine name that does not agree with the `Permissions` file. This error could be an attempt to masquerade.
DEVICE LOCKED	The calling device to be used is currently locked and in use by another process.
ASSERT ERROR	An `ASSERT` error occurred. Check the `/var/uucp/.Admin/errors` file for the error message and refer to the section "UUCP ASSERT Error Messages" on page 204.
SYSTEM NOT IN Systems FILE	The system is not in the `Systems` file.
CAN'T ACCESS DEVICE	The device tried does not exist or the modes are wrong. Check the appropriate entries in the `Systems` and `Devices` files.
DEVICE FAILED	The device could not be opened.

TABLE 12–8 UUCP STATUS Messages *(Continued)*

Error Message	Description/Action
`WRONG MACHINE NAME`	The called machine is reporting a different name than expected.
`CALLBACK REQUIRED`	The called machine requires that it call your machine.
`REMOTE HAS A LCK FILE FOR ME`	The remote machine has a `LCK` file for your machine. The remote machine could be trying to call your machine. If the remote machine has an older version of UUCP, the process that was talking to your machine might have failed, leaving the `LCK` file. If the remote machine has the new version of UUCP and is not communicating with your machine, the process that has a `LCK` file is hung.
`REMOTE DOES NOT KNOW ME`	The remote machine does not have the node name of your machine in its `Systems` file.
`REMOTE REJECT AFTER LOGIN`	The login that was used by your machine to log in does not agree with what the remote machine was expecting.
`REMOTE REJECT, UNKNOWN MESSAGE`	The remote machine rejected the communication with your machine for an unknown reason. The remote machine might not be running a standard version of UUCP.
`STARTUP FAILED`	Login succeeded, but initial handshake failed.
`CALLER SCRIPT FAILED`	This error is usually the same as `DIAL FAILED`. However, if this error occurs often, suspect the caller script in the `Dialers` file. Use `Uutry` to check.

UUCP Numerical Error Messages

The following table lists the exit code numbers of error status messages that are produced by the `/usr/include/sysexits.h` file. Not all are currently used by `uucp`.

TABLE 12–9 UUCP Error Messages by Number

Message Number	Description	Meaning
64	Base Value for Error Messages	Error messages begin at this value.
64	Command–Line Usage Error	The command was used incorrectly, for example, with the wrong number of arguments, a bad flag, or a bad syntax.
65	Data Format Error	The input data was incorrect in some way. This data format should only be used for user's data and not system files.
66	Cannot Open Input	An input file, not a system file, did not exist, or was not readable. This problem could also include errors like “No message” to a mailer.
67	Address Unknown	The user that was specified did not exist. This error might be used for mail addresses or remote logins.
68	Host Name Unknown	The host did not exist. This error is used in mail addresses or network requests.

TABLE 12–9 UUCP Error Messages by Number *(Continued)*

Message Number	Description	Meaning
69	Service Unavailable	A service is unavailable. This error can occur if a support program or file does not exist. This message also can simply indicate that something does not work and the cause currently is not identifiable.
70	Internal Software Error	An internal software error has been detected. This error should be limited to non-operating system-related errors, if possible.
71	System Error	An operating system error has been detected. This error is intended to be used for conditions like "cannot fork", "cannot create pipe." For instance, this error includes a `getuid` return of a user who does not exist in the `passwd` file.
72	Critical OS File Missing	A system file such as `/etc/passwd` or `/var/admin/utmpx` does not exist, cannot be opened, or has an error, such as a syntax error.
73	Can't Create Output File	A user-specified output file cannot be created.
74	Input/Output Error	An error occurred while doing I/O on some file.
75	Temporary Failure. User is invited to retry	Temporary failure that is not really an error. In `sendmail`, this means that a mailer, for example, could not create a connection, and the request should be reattempted later.
76	Remote Error in Protocol	The remote system returned something that was "not possible" during a protocol exchange.
77	Permission Denied	You do not have sufficient permission to perform the operation. This message is not intended for file system problems, which should use `NOINPUT` or `CANTCREAT`, but rather for higher-level permissions. For example, `kre` uses this message to restrict students who can send mail to.
78	Configuration Error	The system detected an error in the configuration.
79	Entry Not Found	Entry not found.
79	Maximum Listed Value	Highest value for error messages.

Index

B

C

D

E

M

N

O

P

T

U

V

W

X