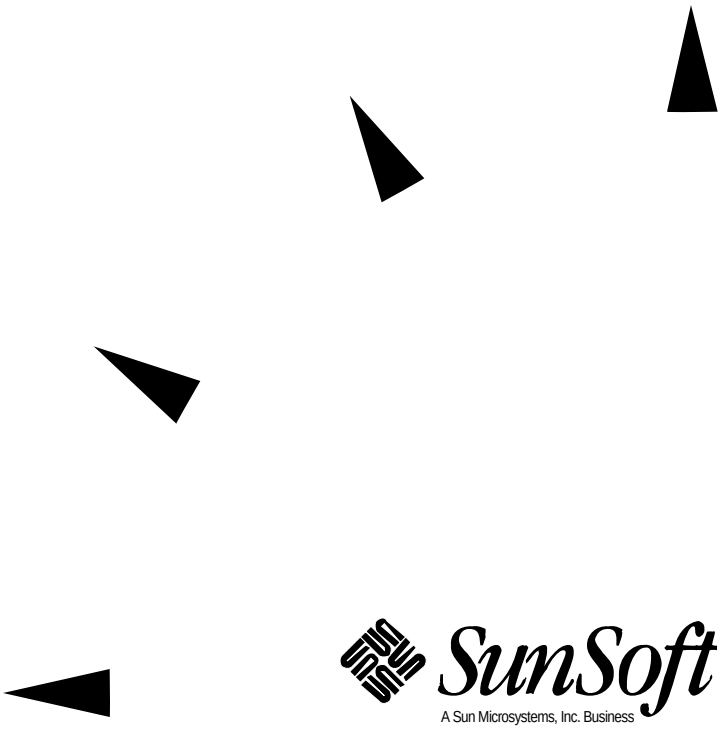
# TCP/IP a. Data Communication

http://kickme.to/tiger/

# TCP/IP and Data Communications Administration Guide



2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

SunSoft
A Sun Microsystems, Inc. Business

© 1995 Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, California 94043-1100 U.S.A.

All rights reserved. This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Portions of this product may be derived from the UNIX® system, licensed from UNIX Systems Laboratories, Inc., a wholly owned subsidiary of Novell, Inc., and from the Berkeley 4.3 BSD system, licensed from the University of California. Third-party software, including font technology in this product, is protected by copyright and licensed from Sun's Suppliers.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19. The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

TRADEMARKS
Sun, Sun Microsystems, the Sun logo, SunSoft, the SunSoft logo, Solaris, SunOS, SunLink,OpenWindows, DeskSet, ONC, ONC+, and NFS are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X / Open Company, Ltd. OPEN LOOK is a registered trademark of Novell, Inc. PostScript and Display PostScript are trademarks of Adobe Systems, Inc.

All SPARC trademarks are trademarks or registered trademarks of SPARC International, Inc. in the United States and other countries. SPARCcenter, SPARCcluster, SPARCompiler, SPARCdesign, SPARC811, SPARCengine, SPARCprinter, SPARCserver, SPARCstation, SPARCstorage, SPARCworks, microSPARC, microSPARC-II, and UltraSPARC are licensed exclusively to Sun Microsystems, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK® and Sun™ Graphical User Interfaces were developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUI's and otherwise comply with Sun's written license agreements.

X Window System is a trademark of X Consortium, Inc.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN, THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND / OR CHANGES IN THE PRODUCT(S) AND / OR THE PROGRAMS(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

Please
Recycle

Adobe PostScript

# *Contents*

Preface . . . . . xxi

*Part 1 —Setting Up and Administering TCP/IP Networks*

**1. Overview of Network Administration . . . . . 1**

Responsibilities of the Network Administrator . . . . . 1

Designing the Network . . . . . 2

Setting Up the Network . . . . . 2

Maintaining the Network . . . . . 2

Expanding the Network . . . . . 3

What TCP / IP Is . . . . . 3

Types of Hardware That Make Up a Solaris Network . . . . 4

How Network Software Transfers Information . . . . . 6

Reaching Beyond the Local-Area Network—the Wide-Area Network . . . . . 9

**2. TCP/IP Protocol Suite . . . . . 11**

Introducing the Internet Protocol Suite . . . . . 11

Protocol Layers and the OSI Model . . . . . 12

TCP/IP Protocol Architecture Model . . . . . 13
How the TCP/IP Protocols Handle Data Communications . . . 19
Data Encapsulation and the TCP/IP Protocol Stack . . . . . 20
Finding Out More About TCP/IP and the Internet . . . . . 24
Computer Trade Books . . . . . 24
RFCs and FYIs . . . . . 24
**3. Planning Your Network . . . . . 27**
Designing the Network . . . . . 27
Factors Involved in Network Planning . . . . . 28
Setting Up an IP Addressing Scheme . . . . . 29
Parts of the IP Address . . . . . 29
Network Classes . . . . . 30
How IP Addresses Apply to Network Interfaces . . . . . 34
Naming Entities on Your Network . . . . . 35
Administering Host Names . . . . . 35
Selecting a Name Service . . . . . 36
Registering Your Network . . . . . 38
InterNIC and InterNIC Registration Services . . . . . 38
How to Contact the InterNIC . . . . . 39
Adding Routers . . . . . 40
Network Topology . . . . . 40
How Routers Transfer Packets . . . . . 42
**4. Configuring TCP/IP on the Network . . . . . 45**
Before You Configure TCP/IP . . . . . 46

Determining Host Configuration Modes . . . . . 47
Machines That Should Run in Local Files Mode . . . . . 47
Machines That Are Network Clients . . . . . 48
Mixed Configurations . . . . . 49
Sample Network . . . . . 49
TCP/IP Configuration Files . . . . . 51
`/etc/hostname.`*interface* File . . . . . 51
`/etc/nodename` File . . . . . 52
`/etc/defaultdomain` File . . . . . 52
`/etc/defaultrouter` File . . . . . 52
`hosts` Database . . . . . 53
`netmasks` Database . . . . . 56
Network Databases and `nsswitch.conf` File . . . . . 60
How Name Services Affect Network Databases . . . . . 61
`nsswitch.conf` File — Specifying Which Name Service to Use . . . . . 63
`bootparams` Database . . . . . 67
`ethers` Database . . . . . 68
Other Network Databases . . . . . 69
`protocols` Database . . . . . 70
`services` Database . . . . . 71
Network Configuration Procedures . . . . . 72
▼ How to Configure a Host for Local Files Mode . . . . . 72
Setting Up a Network Configuration Server . . . . . 74

Configuring Network Clients. . . . . 75
Configuring Standard TCP / IP Services . . . . . 77
Overview of the Booting Processes . . . . . 77
**5. Configuring Routers . . . . . 81**
Routing Protocols. . . . . 81
Routing Information Protocol (RIP) . . . . . 82
ICMP Router Discovery (RDISC) Protocol . . . . . 82
Configuring Routers . . . . . 82
Configuring Both Router Network Interfaces . . . . . 83
How a Machine Determines If It Is a Router . . . . . 84
Automatic Routing Protocol Selection . . . . . 85
Forcing a Machine to Be a Router . . . . . 85
Creating a Multihomed Host . . . . . 86
Turning On Space-Saving Mode . . . . . 87
Turning Off ICMP Router Discovery on the Host . . . . . 87
Turning Off ICMP Router Discovery on the Router . . . . . 87
**6. Troubleshooting TCP/IP . . . . . 89**
General Troubleshooting Methods . . . . . 89
Running Software Checks. . . . . 90
`ping` Command. . . . . 90
`ifconfig` Command . . . . . 92
`netstat` Command . . . . . 93
Displaying Per Protocol Statistics . . . . . 94
Displaying Network Interface Status . . . . . 95

Displaying Routing Table Status . . . . . 96
Logging Network Problems . . . . . 97
Displaying Packet Contents . . . . . 97

*Part 2 —Expanding Your Network With PPP*

**7. Understanding PPP. . . . . 101**
Overview of Solaris PPP . . . . . 101
Solaris PPP Specifications . . . . . 102
Transmission Facilities Used by PPP . . . . . 102
Standards Conformance . . . . . 102
PPP Network Interfaces . . . . . 103
Extending Your Network With PPP. . . . . 103
Point-to-Point Communications Links . . . . . 103
Point-to-Point Configurations Supported by Solaris PPP. . 105
Multipoint Communications Links . . . . . 109
Multipoint Configurations Supported by PPP . . . . . 109
Introducing the PPP Software . . . . . 111
Link Manager . . . . . 112
Login Service . . . . . 113
Configuration File . . . . . 113
Log File . . . . . 114
FIFO File . . . . . 114
UUCP Databases . . . . . 114
How the Components Work Together . . . . . 114

Outbound Connections Scenario . . . . . 114
Inbound Connections Scenario . . . . . 115
PPP Security . . . . . 116
**8. Preparing Your PPP Configuration. . . . . 117**
Determining Requirements for Your Configuration Type . . . . 118
Remote Computer-to-Network Configuration. . . . . 118
Remote Host-to-Remote Host Configuration . . . . . 119
Network-to-Network Configuration . . . . . 120
Dial-in Server With Dynamic Point-to-Point Links . . . . . 121
Multipoint Dial-in Server . . . . . 122
Hosts on a Virtual Network . . . . . 123
Determining IP Addressing for Your PPP Link . . . . . 123
Specifying IP Addresses . . . . . 123
Types of Addressing Schemes . . . . . 124
Routing Considerations . . . . . 126
Turning Off RIP . . . . . 127
PPP Hardware Requirements . . . . . 127
File Space Requirements . . . . . 128
Checklist for Configuring PPP . . . . . 128
**9. Configuring PPP . . . . . 131**
Overview of the Configuration Process . . . . . 131
Installing the PPP Software . . . . . 132
Verifying Installation . . . . . 132
Sample PPP Configuration . . . . . 133

Editing the /etc/inet/hosts File . . . . . 134
▼ How to Configure the Remote Machine's hosts Database 135
Multipoint Dial-in Server hosts Database . . . . . 135
▼ How to Configure the Dial-In Server's hosts Database 136
Editing UUCP Databases . . . . . 136
Updating /etc/uucp/Devices for PPP . . . . . 137
Updating /etc/uucp/Dialers for PPP . . . . . 137
Updating /etc/uucp/Systems for PPP. . . . . 138
Modifying the /etc/passwd File. . . . . 138
Editing the /etc/asppp.cf Configuration File . . . . . 139
Parts of Basic Configuration File . . . . . 140
Configuration File for Multipoint Dial-in Server. . . . . 142
Editing the Configuration File . . . . . 145
Adding PPP Security . . . . . 146
Starting up and Stopping Your New PPP Link . . . . . 146
▼ How to Manually Start PPP . . . . . 146
▼ How to Verify That PPP Is Running . . . . . 146
▼ How to Stop PPP . . . . . 147
**10. Troubleshooting PPP . . . . . 149**
Checking Hardware. . . . . 150
Checking Interface Status . . . . . 150
Checking Connectivity . . . . . 151
Checking Interface Activity . . . . . 151

Checking the Local Routing Tables . . . . . 152
Checking Permissions . . . . . 153
Checking Packet Flow . . . . . 153
Using PPP Diagnostics for Troubleshooting . . . . . 155
▼ How to Set Diagnostics for Your Machine . . . . . 155
Analyzing Diagnostic Output . . . . . 156
**11. Tailoring Your PPP Link . . . . . 165**
Configuring Dynamically Allocated PPP Links . . . . . 165
Addressing Issues for Dynamically Allocated Links . . . . . 167
Updating the `hosts` Database for Dynamic Links . . . . . 167
Considerations for Other Files . . . . . 168
Editing `asppp.cf` for Dynamic Link . . . . . 169
Configuring a Virtual Network . . . . . 172
Addressing Issues for Virtual Networks . . . . . 173
Updating `hosts` and `networks` Databases . . . . . 173
Considerations for Other Files . . . . . 174
`asppp.cf` Configuration File for a Virtual Network . . . . . 174
▼ How to Install PAP/CHAP . . . . . 176
Configuration Keywords . . . . . 180
*Part 3 —Administering UUCP Communications*
**12. UUCP Databases and Programs . . . . . 185**
UUCP Hardware Configurations . . . . . 186
Software Comprising UUCP . . . . . 186
Daemons . . . . . 186

Administrative Programs . . . . . 187
User Programs . . . . . 188
Introducing the UUCP Database Files. . . . . 189
Configuring UUCP Files . . . . . 190
`/etc/uucp/Systems` File . . . . . 191
System-Name Field . . . . . 191
Time Field . . . . . 191
Type Field . . . . . 193
Speed Field . . . . . 193
Phone Field . . . . . 194
Chat-Script Field . . . . . 195
Hardware Flow Control . . . . . 197
Setting Parity . . . . . 198
`/etc/uucp/Devices` File . . . . . 198
Type Field . . . . . 199
Line Field . . . . . 200
Line2 Field . . . . . 201
Class Field. . . . . 201
Dialer-Token-Pairs Field . . . . . 202
Protocol Definitions in the `Devices` File . . . . . 205
`/etc/uucp/Dialers` File . . . . . 206
Hardware Flow Control . . . . . 210
Setting Parity . . . . . 211
Other Basic Configuration Files . . . . . 211

/etc/uucp/Dialcodes File . . . . . 211
/etc/uucp/Sysfiles File . . . . . 213
/etc/uucp/Sysname File . . . . . 214
/etc/uucp/Permissions File . . . . . 214
Structuring Entries . . . . . 215
Considerations . . . . . 215
REQUEST Option . . . . . 216
SENDFILES Option . . . . . 216
MYNAME Option . . . . . 217
READ and WRITE Options . . . . . 217
NOREAD and NOWRITE Options . . . . . 218
CALLBACK Option . . . . . 218
COMMANDS Option . . . . . 219
VALIDATE Option . . . . . 221
MACHINE Entry for OTHER . . . . . 223
Combining MACHINE and LOGNAME . . . . . 223
Forwarding . . . . . 224
/etc/uucp/Poll File . . . . . 224
/etc/uucp/Config File . . . . . 224
/etc/uucp/Grades File . . . . . 225
User-job-grade Field . . . . . 225
System-job-grade Field . . . . . 225
Job-size Field . . . . . 227
Permit-type Field . . . . . 227

ID-list Field . . . . . 228
Other UUCP Configuration Files . . . . . 228
`/etc/uucp/Devconfig` File . . . . . 228
`/etc/uucp/Limits` File . . . . . 228
`remote.unknown` File . . . . . 229
Administrative Files . . . . . 229
**13. Configuring and Maintaining UUCP . . . . . 233**
Adding UUCP Logins . . . . . 233
Starting UUCP . . . . . 234
`uudemon.poll` Shell Script . . . . . 235
`uudemon.hour` Shell Script . . . . . 235
`uudemon.admin` Shell Script . . . . . 235
`uudemon.cleanup` Shell Script . . . . . 236
Running UUCP Over TCP/IP . . . . . 236
Activating UUCP in `/etc/inetd.conf` . . . . . 236
Tailoring Systems File Entries for TCP/IP . . . . . 236
Checking `/etc/inet/services` for UUCP . . . . . 237
Security, Maintenance, and Troubleshooting . . . . . 237
Setting Up UUCP Security . . . . . 237
Regular UUCP Maintenance . . . . . 238
Troubleshooting UUCP . . . . . 239
UUCP Error Messages . . . . . 241
Index . . . . . 249

# *Figures*


Figure 1-1 A Solaris Local Area Network . . . . . 5

Figure 1-2 How Information Is Transferred on a Network . . . . . 6

Figure 2-1 How a Packet Travels Through the TCP/IP Stack . . . . . 20

Figure 3-1 Parts of an IP Address. . . . . 30

Figure 3-2 Byte Assignment in a Class A Address . . . . . 31

Figure 3-3 Byte Assignment in a Class B Address. . . . . 32

Figure 3-4 Byte Assignment in a Class C Address . . . . . 32

Figure 3-5 Basic Network Topology . . . . . 41

Figure 3-6 Providing Additional Path Between Networks . . . . . 42

Figure 3-7 Three Interconnected Networks . . . . . 43

Figure 4-1 Hosts in a Sample Network . . . . . 50

Figure 4-2 Forms of the `hosts` Database Used by Name Services. . . . . 62

Figure 7-1 Basic Point-to-Point Link . . . . . 104

Figure 7-2 Nomadic Computers and Dynamic Link Dial-in Server . . . . . 106

Figure 7-3 Two Networks Connected by a PPP Link . . . . . 108

Figure 7-4 Nomadic Computers and Multipoint Dial-in Server . . . . . 109

Figure 7-5 Virtual Network of Nomadic Computers . . . . . 111
Figure 7-6 PPP Component Software . . . . . 112
Figure 9-1 Sample Network of Remote Hosts and Multipoint Dial-in Server 134
Figure 11-1 Network of Remote Hosts and Dynamic Link Dial-in Servers 166
Figure 11-2 Sample Virtual Network. . . . . 172

# *Tables*


Table 2-1 The Open Systems Interconnect Reference Model . . . . . 13

Table 2-2 TCP/IP Protocol Stack . . . . . 14

Table 3-1 Division of IP Address Space. . . . . 33

Table 3-2 Range of Available Numbers. . . . . 34

Table 4-1 Network Databases and Corresponding Name Service Files. 63

Table 11-1 Authenticator Keywords and Associated Strings. . . . . 176

Table 11-2 Peer Keywords and Associated Strings . . . . . 176

Table 11-3 PAP/CHAP Keyword Definitions . . . . . 178

Table 11-4 Required Keywords for `asppp.cf` . . . . . 181

Table 11-5 Optional Keywords for `asppp.cf`. . . . . 182

Table 12-1 Fields in `/etc/uucp/Systems` . . . . . 191

Table 12-2 Day Field . . . . . 192

Table 12-3 Type Field and `/etc/uucp/Devices` File . . . . . 193

Table 12-5 Phone Field Correspondence. . . . . 194

Table 12-4 Speed Field and `/etc/uucp/Devices` File . . . . . 194

Table 12-6 Escape Characters Used in Systems File Chat Script . . . . . 196

Table 12-7 Type Field and /etc/uucp/Systems File Equivalent ..... 200
Table 12-8 Class Field and /etc/uucp/Systems Correspondence.... 201
Table 12-9 Dialer-Token Pairs..... 202
Table 12-10 Dialers Field and /etc/uucp/Dialers Correspondence.. 203
Table 12-11 Dialers Field Using a Port Selector..... 204
Table 12-12 Dialers Field With Two Dialer-Token-Pairs..... 204
Table 12-13 Protocols Used in /etc/uucp/Devices ..... 206
Table 12-14 /etc/uucp/Dialers File Entry ..... 207
Table 12-15 Backslash Characters for /etc/uucp/Dialers..... 209
Table 12-16 Correspondences between Dialcodes and Systems Files. 212
Table 12-17 Entries in the Dialcodes File ..... 212
Table 12-18 Job-size Field ..... 227
Table 12-19 Permit-type Field ..... 227
Table 12-20 UUCP Lock Files..... 230
Table 13-1 ASSERT Error Messages..... 241
Table 13-2 UUCP STATUS Messages ..... 244
Table 13-3 UUCP Error Messages by Number..... 246

# *Code Samples*


Code Example 4-1 `/etc/inet/hosts` File for Machine `ahaggar` .... 54

Code Example 4-2 `/etc/inet/hosts` File for Machine `timbuktu` ... 55

Code Example 4-3 `/etc/inet/hosts` File for Machine Running in Local Files Mode ..... 56

Code Example 4-4 `/etc/inet/netmasks` File for a Class B Network . 59

Code Example 4-5 `nsswitch.conf` for Networks Using Files for Name Service..... 64

Code Example 4-6 `nsswitch.conf` for a Diskless Client on a Network Running NIS ..... 66

Code Example 4-7 `bootparams` Database ..... 67

Code Example 4-8 Entries in the `ethers` Database ..... 68

Code Example 4-9 `/etc/networks` File..... 70

Code Example 4-10 `/etc/inet/protocols` File..... 71

Code Example 4-11 `/etc/inet/services` File..... 71

Code Example 7-1 An Unmodified `asppp.cf` File ..... 113

Code Example 8-1 `/etc/inet/networks` File for an Internetwork With PPP ..... 126

Code Example 9-1 Basic Configuration File..... 140

Code Example 9-2 Configuration File for a Multipoint Dial-in Server . . 143
Code Example 10-1 `ifconfig` Output for Point-to-Point Link . . . . . 150
Code Example 10-2 `ifconfig` Output for Multipoint Link . . . . . 151
Code Example 10-3 Sample Output from snoop . . . . . 154
Code Example 10-4 Configuration Request . . . . . 160
Code Example 11-1 Configuration File for Server With Dynamically Allocated Link . . . . . 170
Code Example 11-2 Configuration File for `nomada` . . . . . 175
Code Example 11-3 Configuration File for `nomadb` . . . . . 175
Code Example 11-4 Code Example for Server `mojave` . . . . . 179
Code Example 11-5 Code Example for Remote Host `tamerlane` . . . . . 180
Code Example 11-6 Code Example for Remote Host `lawrence` . . . . . 180
Code Example 12-1 Excerpts From `/etc/uucp/Dialers` . . . . . 208

# *Preface*

*TCP/IP and Data Communications Administration Guide* explains how to set up, maintain, and expand a network running the Solaris™ implementation of the TCP/IP protocol suite. The manual

- Defines networking concepts used when working with TCP/IP
- Describes tasks necessary for setting up a new network
- Presents information for maintaining the network
- Explains how to expand the existing network by using routers to create an internetwork
- Describes how to use Point-to-Point Protocol (PPP) to allow remote machines to connect to the network
- Explains how to set up communications with remote machines through the UNIX-to-UNIX Copy Program (UUCP)

## *Who Should Use This Book*

*TCP/IP and Data Communications Administration Guide* contains information for network administrators with a wide range of experience. The text assumes that you are already familiar with the Solaris environment and have administered local machines and peripheral devices such as modems.

If you are setting up a new network, read *TCP/IP and Data Communications Administration Guide* before going on to the other books in the Solaris 2.5 System Administrator set. If you are administering or expanding an existing network, refer to the specific chapters related to the tasks that you want to perform.

**Note –** If you need to set up a brand new network at a site that has never had a Solaris or other UNIX-based network, read Chapter 3, “Planning Your Network,” before installing the Solaris software. This chapter provides important information that supplements the installation tasks in *SPARC: Installing Solaris Software*.

You do not have to read the chapters in sequence, but each chapter assumes that you are familiar with the contents of previous chapter.

## *Before You Read This Book*

You should be familiar with the information contained in the following books before continuing with the *TCP/IP and Data Communications Administration Guide*:

- *SPARC: Installing Solaris Software*
- *x86: Installing Solaris Software*
- *Solaris Advanced User’s Guide*
- *Mail Administration Guide*
- *System Administration Guide, Volume I*
- *System Administration Guide, Volume II*

## *How This Book Is Organized*

*TCP/IP and Data Communications Administration Guide* contains the following chapters.

**Chapter 1, “Overview of Network Administration,**” describes the tasks a network administrator is likely to perform and introduces basic networking concepts.

**Chapter 2, “TCP/IP Protocol Suite**,” introduces the protocols composing the TCP/IP protocol suite.

**Chapter 3, “Planning Your Network,”** presents the issues that you need to consider when designing a new network, such as Internet Protocol (IP) addressing, network topology, and so forth.

**Chapter 4, “Configuring TCP/IP on the Network,”** contains procedures for setting up the machines on the new network.

**Chapter 5, "Configuring Routers,"** explains how to expand the network through use of routers.

**Chapter 6, "Troubleshooting TCP/IP,"** explains how to use the tools available for diagnosing and fixing TCP/IP-related problems.

**Chapter 7, "Understanding PPP,"** introduces the PPP data link protocol that enables you to expand a network through use of modems and phone lines.

**Chapter 8, "Preparing Your PPP Configuration,"** explains issues that you need to consider when designing a particular PPP configuration.

**Chapter 9, "Configuring PPP,"** contains procedures for configuring two basic types of PPP links.

**Chapter 10, "Troubleshooting PPP,"** explains how to diagnose and fix problems related to PPP.

**Chapter 11, "Tailoring Your PPP Link,"** contains information for setting up more complex PPP links.

**Chapter 12, "UUCP Databases and Programs,"** explains how to set up the UUCP database files.

**Chapter 13, "Configuring and Maintaining UUCP,"** explains how to start up UUCP and troubleshoot problems on UUCP links.

## *Related Books*

After you have set up the network, you probably will want to add the network services provided by the Solaris operating system. They are described in the following books that are part of your System Administration Document Set:

- *NFS Administration Guide*
- *NIS+ and FNS Administration Guide*
- *NIS+ and DNS Setup and Configuration Guide*
- *NIS+ Transition Guide*

You can also find invaluable information for managing heterogeneous TCP/IP networks in the following books:

- *UNIX Communications* by Bart Anderson, Bryan Costales and Harry Henderson. (Howard W. Sams & Company, 1987)

- *Firewalls and Internet Security* by William R. Cheswick and Steven M. Bellovin. (Addison Wesley, 1994)
- *TCP/IP Network Administration* by Craig Hunt. (O' Reilly & Associates, Inc., 1993).
- *The Whole Internet User's Guide and Catalog* by Ed Krol.( O' Reilly & Associates, Inc., 1993.)
- *Managing UUCP and Usenet* by Tim O' Reilly and Grace Todino. (O' Reilly & Associates, Inc., 1992.)
- *TCP/IP Illustrated, Volume 1, The Protocols* by W. Richard Stevens . (Addison Wesley, 1994.)

## *What Typographic Changes and Symbols Mean*

The following table describes the type changes and symbols used in this book

*Table P-1* Typographic Conventions

| Typeface or Symbol | Meaning | Example |
|---|---|---|
| `AaBbCc123` | The names of commands, files, and directories; on-screen computer output | Edit your `.login` file.<br>Use `ls -a` to list all files.<br>`machine_name% You have mail.` |
| **`AaBbCc123`** | What you type, contrasted with on-screen computer output | `machine_name% su`<br>`Password:` |
| *AaBbCc123* | Command-line placeholder: replace with a real name or value | To delete a file, type `rm` *filename.* |
| *AaBbCc123* | Book titles, new words or terms, or words to be emphasized | Read Chapter 6 in *User's Guide.*<br>These are called *class* options.<br>You *must* be root to do this. |

## *Shell Prompts in Command Examples*

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

*Table P-2* Shell Prompts

| **Shell** | **Prompt** |
|---|---|
| C shell prompt | `machine_name%` |
| C shell superuser prompt | `machine_name#` |
| Bourne shell and Korn shell prompt | `$` |
| Bourne shell and Korn shell superuser prompt | # |

# *Part 1 — Setting Up and Administering TCP/IP Networks*

Part 1 explains how to set up a network that will run the Solaris implementation of TCP/IP. The text assumes that you are familiar with UNIX and have some experience administering local UNIX systems. It does not assume that you are an experienced network administrator.


| | |
|---|---|
| *Responsibilities of the Network Administrator* | *page 1* |
| *Types of Hardware That Make Up a Solaris Network* | *page 4* |
| *Introducing the Internet Protocol Suite* | *page 20* |
| *Designing Your IP Addressing Scheme* | *page 33* |
| *Selecting a Name Service* | *page 36* |
| *Determining Host Configuration Modes* | *page 47* |
| *TCP/IP Configuration Files* | *page 51* |
| *Network Configuration Procedures* | *page 72* |
| *Routing Protocols* | *page 81* |
| *Configuring Routers* | *page 82* |
| *General Troubleshooting Methods* | *page 89* |
| *Running Software Checks* | *page 90* |

# *Overview of Network Administration* 1

This chapter introduces the role of the network administrator. If you are a new network administrator, the topics covered will give you an idea of the tasks you may perform. The chapter then presents fundamental networking concepts that you'll need to know as you progress through this book. If you are an experienced network administrator, consider moving on to the next chapter.


| | |
|---|---|
| *Designing the Network* | *page 2* |
| *Maintaining the Network* | *page 2* |
| *Expanding the Network* | *page 3* |
| *What TCP/IP Is* | *page 3* |
| *Types of Hardware That Make Up a Solaris Network* | *page 4* |
| *How Information Is Transferred: The Packet* | *page 7* |
| *Who Sends and Receives Information: The Host* | *page 7* |


## *Responsibilities of the Network Administrator*

As a network administrator, your tasks generally will fall into four areas:

- Designing and planning the network
- Setting up the network
- Maintaining the network
- Expanding the network

Each task area corresponds to a phase in the continuing life cycle of a network. You may be responsible for all the phases, or you may ultimately specialize in a particular area; for example, network maintenance.

## *Designing the Network*

The first phase in the life cycle of a network involves creating its design, a task not usually performed by new network administrators. Designing a network involves making decisions about the type of network that best suits the needs of your organization. In larger sites this task is performed by a senior network architect: an experienced network administrator familiar with both network software and hardware.

Chapter 3, “Planning Your Network,” describes the factors involved in network design.

## *Setting Up the Network*

After the new network is designed, the second phase of network administration begins, which involves setting up and configuring the network. This consists of installing the hardware that makes up the physical part of the network, and configuring the files or databases, hosts, routers, and network configuration servers.

The tasks involved in this phase are a major responsibility for network administrators. You should expect to perform these tasks unless your organization is very large, with an adequate network structure already in place.

Chapter 4, “Configuring TCP/IP on the Network,” contains instructions for the tasks involved in this phase of the network life cycle.

## *Maintaining the Network*

The third phase of network administration consists of ongoing tasks that will typically comprise the bulk of your responsibilities. They might include

- Adding new host machines to the network
- Network security
- Administering network services, such as NFS®, name services, and electronic mail

- Troubleshooting network problems

Chapter 4 explains how to set up new hosts on an existing network. Chapter 6, "Troubleshooting TCP/IP," contains hints for solving network problems. For information on network services, refer to *NFS Administration Guide, NIS+ and FNS Administration Guide, NIS+ Transition Guide*, and *Mail Administration Guide*. For security-related tasks, refer to *System Administration Guide, Volume I*.

## *Expanding the Network*

The longer a network is in place and functioning properly, the more your organization will want to expand its features and services. Initially, you can increase network population by adding new hosts and expand network services by providing additional shared software. But eventually, a single network will expand to the point where it can no longer operate efficiently. That is when it must enter the fourth phase of the network administration cycle: expansion.

Several options are available for expanding your network:

- Setting up a new network and connecting it to the existing network using a machine functioning as a router, thus creating an internetwork.
- Configuring machines in users' homes or in remote office sites and enabling these machines to connect over telephone lines to your network.
- Connecting your network to the Internet, thus enabling users on your network to retrieve information from other systems throughout the world.
- Configuring UUCP communications, enabling users to exchange files and electronic mail with remote machines.

Chapter 5, "Configuring Routers," contains procedures for setting up an internetwork. Part 2, "Expanding Your Network with PPP," explains how to set up networking connections for nomadic computers. Part 3, "Administering UUCP Communications," explains how to use UUCP to exchange information between your machine and other UUCP systems.

## *What TCP/IP Is*

A network *communications protocol* is a set of formal rules that describe how software and hardware should interact within a network. For the network to function properly, information must be delivered to the intended destination

in an intelligible form. Because different types of networking software and hardware need to interact to perform the network function, designers developed the concept of the communications protocol.

The Solaris operating system includes the software needed for network operations for your organization. This networking software implements the communications protocol suite collectively referred to as *TCP/IP*. TCP/IP is recognized as a standard by major international standards organizations and is used throughout the world. Because it is a set of standards, TCP/IP runs on many different types of computers, making it easy for you to set up a heterogeneous network running the Solaris operating system.

TCP/IP provides services to many different types of computers, operating systems, and networks. Types of networks range from local area networks, such as Ethernet, FDDI, and Token Ring, to wide-area networks, such as T1 (telephone lines), X.25, and ATM.

You can use TCP/IP to construct a network out of a number of local-area networks. You can also use TCP/IP to construct a wide-area network by way of virtually any point-to-point digital circuit.

TCP/IP and its protocol family are fully described in Chapter 2, "TCP/IP Protocol Suite."

## *Types of Hardware That Make Up a Solaris Network*

The term local-area network (LAN) refers to a single network of computers limited to a moderate geographical range, such as the floor of a building or two adjacent buildings. A local-area network has both hardware and software components. From a hardware perspective, a basic Solaris LAN consists of two or more computers attached to some form of local-area network media.

### *Local-Area Network Media*

The cabling or wiring used for computer networks is referred to as *network media*. Figure 1-1 shows four computers connected by means of Ethernet media. In the Solaris LAN environment, Ethernet is the most commonly used local-area network media. Other types of local-area network media used in a Solaris LAN might include FDDI or Token Ring.

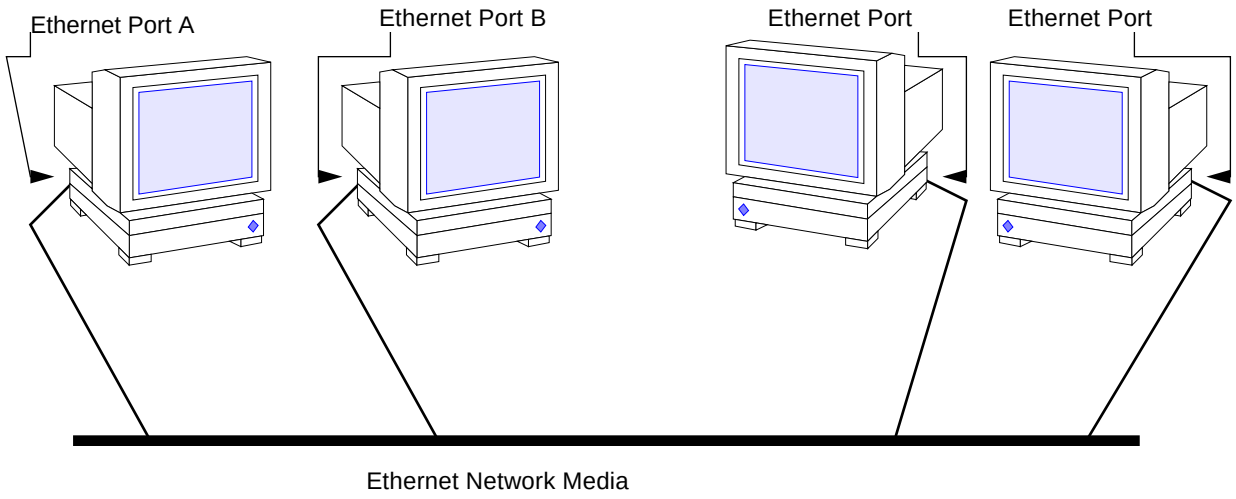



*Figure 1-1* A Solaris Local Area Network

## *Computers and Their Connectors*

Computers on a TCP/IP network use two different kinds of connectors to connect to network media: serial ports, and the ports on the network interface.

### *Serial Ports*

Each computer has at least two *serial ports*, the connectors that enable you to plug a printer or modem into the computer. The serial ports may be attached to the CPU board, or you may have to purchase them. You use these ports when attaching a modem to the system to establish a PPP or UUCP connection. PPP and UUCP actually provide wide-area network services, since they may use telephone lines as their network media.

### *Network Interfaces*

The hardware in a computer that enables you to connect it to a network is known as a *network interface*. Many computers come with a preinstalled network interface; others may require you to purchase the network interface separately.

Each LAN media type has its own associated network interface. For example, if you want to use Ethernet as your network media, you must have an Ethernet interface installed in each host to be part of the network. The connectors on the board to which you attach the Ethernet cable are referred to as *Ethernet ports*. If you plan to use FDDI, each prospective host must have an FDDI network interface, and so on.

This book refers to the default network interface on a host as the *primary network interface.*

---

**Note –** Installing network hardware is outside the scope of this guide. Refer to *System Administration Guide, Volume II* for instructions for configuring serial ports and manuals accompanying network media for installation instructions.

---

## *How Network Software Transfers Information*

Setting up network software is an involved task. Therefore, it helps to understand how the network software you are about to set up will transfer information.

Figure 1-2 shows the basic elements involved in network communication.

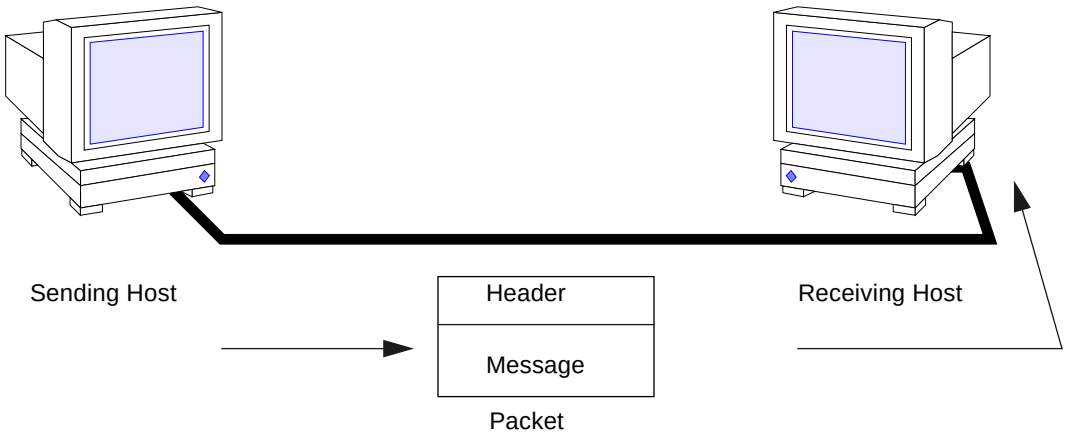



*Figure 1-2* How Information Is Transferred on a Network

In this figure, a computer sends a packet over the network media to another computer attached to the same media.

## *How Information Is Transferred: The Packet*

The basic unit of information to be transferred over the network is referred to as a *packet*. A packet is organized much like a conventional letter.

Each packet has a *header*, which corresponds to the envelope. The header contains the addresses of the recipient and the sender, plus information on how to handle the packet as it travels through each layer of the protocol suite.

The *message* part of the packet corresponds to the letter itself. Packets can only contain a finite number of bytes of data, depending on the network media in use. Therefore, typical communications such as email messages are sometimes split into packet fragments.

## *Who Sends and Receives Information: The Host*

If you are an experienced Solaris user, you are no doubt familiar with the term “host,” a word often used as a synonym for “computer” or “machine.” From a TCP/IP perspective, only two types of entities exist on a network: routers and hosts.

A *router* is a machine that forwards packets from one network to another. To do this, the router must have at least two network interfaces. A machine with only one network interface cannot forward packets; it is considered a *host*. Most of the machines you set up on a network will be hosts.

It is possible for a machine to have more than one network interface but not function as a router. This type of machine is called a *multihomed* host. A multihomed host is directly connected to multiple networks through its network interfaces.However, it does not route packets from one network to another.

When a host initiates communication, it is called a *sending* host, or, simply, the sender. For example, a host initiates communications when its user types `rlogin` or sends an email message to another user. The host that is the target of the communication is called the *receiving* host, or recipient. For example, the remote host specified as the argument to `rlogin` is the recipient of the request to log in.

# 1

Each host has three characteristics that help identify it to its peers on the network. These characteristics include its

- Host name
- Internet address, or *IP address*, the form used in this book
- Hardware address

### *Host Name*

The *host name* is the name of the local machine, combined with the name of your organization. Many organizations let users choose the host names for their machines. Programs such as `sendmail` and `rlogin` use host names to specify remote machines on a network. *System Administration Guide, Volume I* contains more information about host names.

The host name of the machine also becomes the name of the primary network interface. This concept becomes important when you set up the network databases or configure routers.

When setting up a network, you will have to obtain the host names of all machines to be involved. You will use this information when setting up network databases, as described in Chapter 4, “Configuring TCP/IP on the Network.”

### *IP Address*

The *IP address* is one of the two types of addresses each machine has on a TCP/IP network that identifies the machine to its peers on the network. This address also gives peer hosts a notion of *where* a particular host is located on the network. If you have installed the Solaris operating environment on a machine on a network, you may recall specifying the IP address during the installation process. IP addressing is a significant aspect of TCP/IP and is explained fully in ”Designing Your IP Addressing Scheme” on page 33.

### *Hardware Address*

Each host on a network has a hardware address, which also identifies it to its peers. This address is physically assigned to the machine’s CPU or network interface by the manufacturer. Each hardware address is unique.

This book uses the term *Ethernet address* to correspond to the hardware address. Because Ethernet is the most commonly used network media on Solaris-based networks, the text assumes that the hardware address of your Solaris host is an Ethernet address. If you are using other network media, such as FDDI, refer to the documentation that came with your media for hardware addressing information.

## *Reaching Beyond the Local-Area Network—the Wide-Area Network*

As your network continues to function successfully, users may need to access information available from other companies, institutes of higher learning, and other organizations not on your LAN. To obtain this information, they may need to communicate over a *wide-area network* (WAN), a network that covers a potentially vast geographic area and uses network media such as leased data or telephone lines, X.25, and ISDN services.

A prime example of a WAN is the Internet, the global public network that is the successor to the WANs for which TCP/IP was originally developed. Other examples of WANs are *enterprise networks*, linking the separate offices of a single corporation into one network spanning an entire country, or perhaps an entire continent. It is entirely possible for your organization to construct its own WAN.

As network administrator, you may have to provide access to WANs to the users on your local net. Within the TCP/IP and UNIX community, the most commonly used public network has been the Internet. Information about directly connecting to the Internet is outside the scope of this book. You can find many helpful books on this subject in a computer bookstore.

### *Security*

Connecting a LAN to a WAN poses some security risks. You will want to make sure your network is protected from unauthorized use, and control access to data and resources. An overview of security issues is provided in *System Administration Guide, Volume II*. Further help can be found in *Firewalls and Internet Security* by William R. Cheswick and Steven M Bellovin (Addison Wesley, 1994).

You can also become informed by subscribing to majordomo@greatcircle.com, citing `subscribe firewalls` in the text. If you prefer the shorter version, cite `firewalls_digest` in the text.

# 1

# TCP/IP Protocol Suite 2

This chapter introduces the Solaris implementation of the TCP/IP network protocol suite. The information is particularly geared to network administrators who are unfamiliar with the TCP/IP. (For an introduction to basic network concepts, see Chapter 1, “Overview of Network Administration.”) If you are an experienced TCP/IP network administrator, consider moving on to chapters covering the tasks you want to perform.


| | |
|---|---|
| *Protocol Layers and the OSI Model* | *page 12* |
| *TCP/IP Protocol Architecture Model* | *page 13* |
| *Standard TCP/IP Services* | *page 17* |
| *Data Encapsulation and the TCP/IP Protocol Stack* | *page 20* |


## Introducing the Internet Protocol Suite

This section presents an in-depth introduction to the protocols that compose TCP/IP. Although the information is purely conceptual, you should learn the names of the protocols and what each does. This is important because TCP/IP books explain tasks with the assumption that you understand the concepts introduced here.

TCP/IP is the commonly used nickname for the set of network protocols composing the *Internet Protocol suite*. Many texts use the term “Internet” to describe both the protocol suite and the global wide-area network. In this book, the “TCP/IP” refers specifically to the Internet protocol suite; “Internet” refers to the wide-area network and the bodies that govern it.

To interconnect your TCP/IP network with other networks, you must obtain a unique IP network number. At the time of this writing, IP network numbers are assigned by an organization known as the InterNIC.

If hosts on your network are going to participate in the Internet Domain Name system (DNS), you must obtain and register a unique domain name. The InterNIC also handles the registration of domain names under certain top-level domains such as .com (commercial), .edu (education), and .gov (government). Chapter 3, "Planning Your Network," contains more information about the InterNIC. (For more information on DNS, refer to *NIS+ and FNS Administration Guide*.)

## Protocol Layers and the OSI Model

Most network protocol suites are structured as a series of layers, sometimes referred to collectively as a *protocol stack*. Each layer is designed for a specific purpose and exists on both the sending and receiving hosts. Each is designed so that a specific layer on one machine sends or receives exactly the same object sent or received by its *peer process* on another machine. These activities take place independently from what is going on in layers above or below the layer under consideration. In other words, each layer on a host acts independently of other layers on the same machine, and in concert with the same layer on other hosts.

### OSI Reference Model

That most network protocol suites are viewed as structured in layers is a result of the Open Systems Interconnect (OSI) Reference Model designed by the International Standards Organization (ISO). The OSI model describes network activities as having a structure of seven layers, each of which has one or more protocols associated with it. The layers represent data transfer operations common to all types of data transfers among cooperating networks.

The protocol layers of the OSI Reference Model are traditionally listed from the top (layer 7) to the bottom (layer 1) up, as shown in Table 2-1 on page 13:

*Table 2-1* The Open Systems Interconnect Reference Model

| Layer No. | Layer Name | Description |
|---|---|---|
| 7 | Application | Consists of standard communication services and applications that everyone can use. |
| 6 | Presentation | Ensures that information is delivered to the receiving machine in a form that it can understand. |
| 5 | Session | Manages the connections and terminations between cooperating computers. |
| 4 | Transport | Manages the transfer of data and assures that received and transmitted data are identical. |
| 3 | Network | Manages data addressing and delivery between networks. |
| 2 | Data Link | Handles the transfer of data across the network media. |
| 1 | Physical | Defines the characteristics of the network hardware. |

The operations defined by the OSI model are purely conceptual and not unique to any particular network protocol suite. For example, the OSI network protocol suite implements all seven layers of the OSI Reference Model. TCP/IP uses some of OSI model layers and combines others. Other network protocols, such as SNA, add an eighth layer.

## *TCP/IP Protocol Architecture Model*

The OSI model describes an idealized network communications protocol family. TCP/IP does not correspond to this model directly, as it either combines several OSI layers into a single layer, or does not use certain layers at all. Table 2-2 on page 14 shows the layers of the Solaris implementation of TCP/IP, listed from topmost layer (application) to lowest (physical network).

.

*Table 2-2* TCP/IP Protocol Stack

| OSI Ref. Layer No. | OSI Layer Equivalent | TCP/IP Layer | TCP/IP Protocol Examples |
|---|---|---|---|
| 5,6,7 | Application, Session, Presentation | Application | NFS, NIS+, DNS, `telnet`, `ftp`, "r" commands[1], RIP, RDISC, SNMP, others |
| 4 | Transport | Transport | TCP, UDP |
| 3 | Network | Internet | IP, ARP, ICMP |
| 2 | Data Link | Data Link | PPP, IEEE 802.2 |
| 1 | Physical | Physical Network | Ethernet (IEEE 802.3) Token Ring, RS-232, others |

1. "r" commands include `rlogin`, `rsh` and `rcp`.

The table shows the TCP/IP protocol layers, their OSI Model equivalents, and examples of the protocols available at each level of the TCP/IP protocol stack. Each host involved in a communication transaction runs its own implementation of the protocol stack.

## *Physical Network Layer*

The physical network layer specifies the characteristics of the hardware to be used for the network. For example, it specifies the physical characteristics of the communications media. The physical layer of TCP/IP describes hardware standards such as IEEE 802.3, the specification for Ethernet network media, and RS-232, the specification for standard pin connectors.

## *Data-Link Layer*

The data-link layer identifies the network protocol type of the packet, in this case TCP/IP. It also provides error control and "framing." Examples of data-link layer protocols are Ethernet IEEE 802.2 framing and Point-to-Point Protocol (PPP) framing.

## *Internet Layer*

This layer, also known as the network layer, accepts and delivers packets for the network. It includes the powerful Internet protocol (IP), the ARP protocol, and the ICMP protocol.

### *IP Protocol*

The IP protocol and its associated routing protocols are possibly the most significant of the entire TCP/IP suite. IP is responsible for:

- *IP addressing* - The IP addressing conventions are part of the IP protocol. (Chapter 3, “Planning Your Network,” describes IP addressing in complete detail.)
- *Host-to-host communications* - IP determines the path a packet must take, based on the receiving host’s IP address.
- *Packet formatting* - IP assembles packets into units known as *IP datagrams*. Datagrams are fully described on page 22.
- *Fragmentation* - If a packet is too large for transmission over the network media, IP on the sending host breaks the packet into smaller fragments. IP on the receiving host then reconstructs the fragments into the original packet.

### *ARP Protocol*

The Address Resolution Protocol (ARP) conceptually exists between the data link and Internet layers. ARP assists IP in directing datagrams to the appropriate receiving host by mapping Ethernet addresses (48 bits long) to known IP addresses (32 bits long).

### *ICMP Protocol*

Internet Control Message Protocol (ICMP) is the protocol responsible for detecting network error conditions and reporting on them. ICMP reports on

- Dropped packets (when packets are arriving too fast to be processed)
- Connectivity failure (when a destination host can’t be reached)
- Redirection (which tells a sending host to use another router)

Chapter 6, “Troubleshooting TCP/IP,” contains more information on the operating system commands that use ICMP for error detection.

## Transport Layer

The TCP/IP transport layer protocols ensure that packets arrive in sequence and without error, by swapping acknowledgments of data reception, and retransmitting lost packets. This type of communication is known as "end-to-end." Transport layer protocols at this level are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

### TCP Protocol

TCP enables applications to communicate with each other as though connected by a physical circuit. TCP sends data in a form that appears to be transmitted in a character-by-character fashion, rather than as discreet packets. This transmission consists of a starting point, which opens the connection, the entire transmission in byte order, and an ending point, which closes the connection.

TCP attaches a header onto the transmitted data. This header contains a large number of parameters that help processes on the sending machine connect to peer processes on the receiving machine.

TCP confirms that a packet has reached its destination by establishing an end-to-end connection between sending and receiving hosts. TCP is therefore considered a "reliable, connection-oriented" protocol.

### UDP Protocol

UDP, the other transport layer protocol, provides datagram delivery service. It does not provide any means of verifying that connection was ever achieved between receiving and sending hosts. Because UDP eliminates the processes of establishing and verifying connections, applications that send small amounts of data use it rather than TCP.

## Application Layer

The application layer defines standard Internet services and network applications that anyone can use. These services work with the transport layer to send and receive data. There are many applications layer protocols, some of which you probably already use. Some of the protocols include:

- Standard TCP/IP services such as the `ftp`, `tftp`, and `telnet` commands
- UNIX "r" commands, such as `rlogin` and `rsh`

- Name services, such as NIS+ and Domain Name System (DNS)
- File services, such as NFS
- SNMP, which enables network management
- RIP and RDISC routing protocols

### *Standard TCP/IP Services*

- ***FTP and Anonymous FTP*** - The File Transfer Protocol (FTP) transfers files to and from a remote network. The protocol includes the `ftp` command (local machine) and the `in.ftpd` daemon (remote machine). FTP lets a user specify the name of the remote host and file transfer command options on the local host's command line. The `in.ftpd` daemon on the remote host then handles the requests from the local host. Unlike `rcp`, `ftp` works even when the remote computer does not run a UNIX-based operating system. A user must log in to the remote computer to make an `ftp` connection unless it has been set up to allow anonymous FTP.

  You can now obtain a wealth of materials from *anonymous FTP servers* connected to the Internet. These servers are set up by universities and other institutions to make certain software, research papers, and other information available to the public domain. When you log in to this type of server, you use the login name `anonymous`, hence the term "anonymous FTP servers."

  Using anonymous FTP and setting up anonymous FTP servers is outside the scope of this manual. However, many trade books, such as *The Whole Internet User's Guide & Catalog*, discuss anonymous FTP in detail. Instructions for using FTP to reach standard machines are in *System Administration Guide, Volume II*. The `ftp`(1) man page describes all `ftp` command options, including those invoked through the command interpreter. The `ftpd`(1M) man page describes the services provided by the daemon `in.ftpd`.

- ***Telnet*** - The Telnet protocol enables terminals and terminal oriented processes to communicate on a network running TCP/IP. It is implemented as the program `telnet` (on local machines) and the daemon `in.telnet` (on remote machines). Telnet provides a user interface through which two hosts can communicate on a character-by-character or line-by-line basis. The application includes a set of commands that are fully documented in the `telnet`(1) man page.

- ***TFTP*** - The trivial file transfer protocol (`tftp`) provides functions similar to `ftp`, but it does not establish `ftp`'s interactive connection. As a result, users cannot list the contents of a directory or change directories. This means that a user must know the full name of the file to be copied. The `tftp`(1) man page describes the `tftp` command set.

### *UNIX "r" Commands*

The UNIX "r" commands enable users to issue commands on their local machines that are actually carried out on the remote host that they specify. These commands include

- `rcp`
- `rlogin`
- `rsh`

Instructions for using these commands are in *Solaris Advanced User's Guide* and in `rcp` (1), `rlogin` (1), and `rsh` (1) man pages.

### *Name Services*

Two name services are available from the Solaris implementation of TCP/IP: NIS+ and DNS.

- ***NIS+*** - NIS+ provides centralized control over network administration services, such as mapping host names to IP and Ethernet addresses, verifying passwords, and so on. See *NIS+ and FNS Administration Guide* for complete details.
- ***Domain Name System*** - The Domain Name System (DNS) provides host names to the IP address service. It also serves as a database for mail administration. For a complete description of this service, see *NIS+ and FNS Administration Guide*. See also the `in.named`(1M) man page.

### *File Services*

The NFS application layer protocol provides file services for the Solaris operating system. You will find complete information about the NFS service in *NFS Administration Guide*.

### *Network Administration*

The Simple Network Management Protocol (SNMP) enables you to view the layout of your network, view status of key machines, and obtain complex network statistics from graphical user interface based software.Many companies offer network management packages that implement SNMP; SunNet Manager™ software is an example.

### *Routing Protocols*

The Routing Information Protocol (RIP) and the Router Discovery Protocol (RDISC) are two routing protocols for TCP/IP networks. They are described in Chapter 5, “Configuring Routers.”

## *How the TCP/IP Protocols Handle Data Communications*

When a user issues a command that uses a TCP/IP application layer protocol, a chain of events is set in motion. The user’s command or message passes through the TCP/IP protocol stack on the local machine, and then across the network media to the protocols on the recipient. The protocols at each layer on the sending host add information to the original data.

As the user’s command makes its way through the protocol stack, protocols on each layer of the sending host also interact with their peers on the receiving host. Figure 2-1 on page 20 shows this interaction.

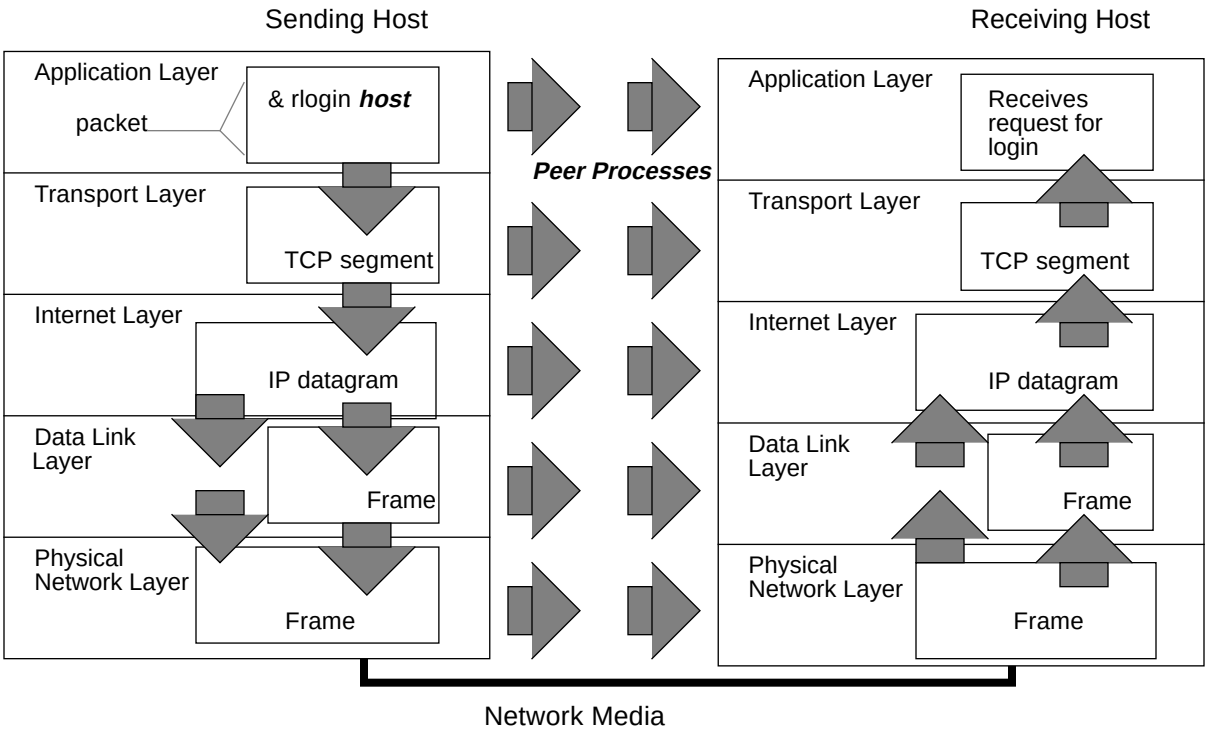



*Figure 2-1* How a Packet Travels Through the TCP/IP Stack

## *Data Encapsulation and the TCP/IP Protocol Stack*

The packet is the basic unit of information transferred across a network, consisting, at aminimum, of a header with the sending and receiving hosts' addresses, and a body with the data to be transferred. As the packet travels through the TCP/IP protocol stack, the protocols at each layer either add or remove fields from the basic header. When a protocol on the sending host adds data to the packet header, the process is called *data encapsulation*. Moreover, each layer has a different term for the altered packet, as shown in Figure 2-1.

This section summarizes the life cycle of a packet from the time the user issues a command or sends a message to the time it is received by the appropriate application on the receiving host.

## *Application Layer—User Initiates Communication*

The packet's history begins when a user on one host sends a message or issues a command that must access a remote host. The application protocol associated with the command or message formats the packet so that it can be handled by the appropriate transport layer protocol, TCP or UDP.

Suppose the user issues an `rlogin` command to log in to the remote host, as shown in Figure 2-1 on page 20. The `rlogin` command uses the TCP transport layer protocol. TCP expects to receive data in the form of a stream of bytes containing the information in the command. Therefore, `rlogin` sends this data as a TCP stream.

Not all application layer protocols use TCP, however. Suppose a user wants to mount a file system on a remote host, thus initiating the NIS+ application layer protocol. NIS+ uses the UDP transport layer protocol. Therefore, the packet containing the command must be formatted in a manner that UDP expects. This type of packet is referred to as a *message*.

## *Transport Layer —Data Encapsulation Begins*

When the data arrives at the transport layer, the protocols at the layer start the process of data encapsulation. The end result depends on whether TCP or UDP has handled the information.

### *TCP Segmentation*

TCP is often called a "connection-oriented" protocol because it ensures the successful delivery of data to the receiving host. Figure 2-1 on page 20 shows how the TCP protocol receives the stream from the `rlogin` command. TCP divides the data received from the application layer into segments and attaches a header to each segment.

Segment headers contain sender and recipient ports, segment ordering information, and a data field known as a *checksum*. The TCP protocols on both hosts use the checksum data to determine whether data has transferred without error.

### *Establishing a TCP Connection*

In addition, TCP uses segments to determine whether the receiving host is ready to receive the data. When the sending TCP wants to establish connections, it sends a segment called a SYN to the peer TCP protocol running on the receiving host. The receiving TCP returns a segment called an ACK to acknowledge the successful receipt of the segment. The sending TCP sends another ACK segment and then proceeds to send the data. This exchange of control information is referred to as a *three-way handshake*.

### *UDP Packets*

UDP is a “connectionless” protocol. Unlike TCP, it does not check to make sure that data arrived at the receiving host. Instead, UDP takes the message received from the application layer and formats it into *UDP packets*. UDP attaches a header to each packet, which contains the sending and receiving host ports, a field with the length of the packet, and a checksum.

The sending UDP process attempts to send the packet to its peer UDP process on the receiving host. The application layer determines whether the receiving UDP process acknowledges that the packet was received. UDP requires no notification of receipt. UDP does not use the three-way handshake.

## *Internet Layer*

As shown in Figure 2-1 on page 20, both TCP and UDP pass their segments and packets down to the Internet layer, where they are handled by the IP protocol. IP prepares them for delivery by formatting them into units called IP datagrams. Then IP determines the IP addresses for the datagrams, so they can effectively be delivered to the receiving host.

### *IP Datagrams*

IP attaches an *IP header* to the segment or packet’s header in addition to the information added by TCP or UDP. Information in the IP header includes the IP addresses of the sending and receiving hosts, datagram length, and datagram sequence order. This is provided in case the datagram exceeds the allowable byte size for network packets and must be fragmented.

### *Data-Link Layer—Framing Takes Place*

Data-link layer protocols such as PPP format the IP datagram into a *frame*. They attach a third header and a footer to “frame” the datagram. The frame header includes a cyclical redundancy check (CRC) field that checks for errors as the frame travels over the network media. Then the data-link layer passes the frame to the physical layer.

### *Physical Network Layer—Preparing the Frame for Transmission*

The physical network layer on the sending host receives the frames and converts the IP addresses into the hardware addresses appropriate to the network media. The physical network layer then sends the frame out over the network media.

### *How the Receiving Host Handles the Packet*

When the packet arrives on the receiving host, it travels through the TCP/IP protocol stack in the reverse order from that which it took on the sender. Figure 2-1 on page 20 illustrates this path. Moreover, each protocol on the receiving host strips off header information attached to the packet by its peer on the sending host. Here is what happens.

1. Physical Network Layer
   Receives the packet in its frame form. It computes the CRC of the packet and then sends the frame to the data link layer.

2. Data-Link Layer
   Verifies that the CRC for the frame is correct and strips off the frame header and CRC. Finally, the data link protocol sends the frame to the Internet layer.

3. Internet Layer
   Reads information in the header to identify the transmission and determine if it is a fragment. If the transmission was fragmented, IP reassembles the fragments into the original datagram. It then strips off the IP header and passes the datagram on to transport layer protocols

4. Transport Layer (TCP and UDP)
   Reads the header to determine which application layer protocol must receive the data. Then TCP or UDP strips off its related header and sends the message or stream up to the receiving application.

5. Application Layer
   Receives the message and performs the operation requested by the sending host.

## Finding Out More About TCP/IP and the Internet

A great deal of information about TCP/IP and the Internet has been published. If you require specific information that is not covered in this text, you probably can find what you need in the sources cited below.

### Computer Trade Books

Many books about TCP/IP and the Internet are available from your local library or computer bookstore. Three highly recommended books are

- *TCP/IP Network Administration* by Craig Hunt - This book contains some theory and much practical information for managing a heterogeneous TCP/IP network.
- *TCP/IP Illustrated, Volume I* by W. Richard Stevens - This book is an in-depth explanation of the TCP/IP protocols.It is ideal for network administrators requiring a technical background in TCP/IP and for network programmers.
- *The Whole Internet User's Guide & Catalog* by Ed Krol - This book is ideal for anyone interested in using the many tools for retrieving information over the Internet.

### RFCs and FYIs

Since 1969, developers working on the Internet protocol suite have described their protocols and related subjects in documents known as Requests for Comments (RFCs). Many RFCs are specifications for particular TCP/IP protocols and describe standards with which software implementing the protocols must comply. Other RFCs discuss the Internet, its topology, and its governing bodies. Still other RFCs explain how to manage TCP/IP applications, such as DNS.

All RFCs must be approved by the Internet Architecture Board (IAB) before they are placed in the public domain. Typically, the information in RFCs is geared to developers and other highly technical readers, though this isn't always the case. Over 1700 RFCs have been published as of this writing.

In recent years, For Your Information (FYI) documents have appeared as a subset of the RFCs. The FYIs contain information that does not deal with Internet standards; rather, they contain Internet information of a more general nature. For example, FYI documents include a bibliography listing introductory TCP/IP books and papers, an exhaustive compendium of Internet-related software tools, and a glossary of Internet and general networking terms.

You'll find references to relevant RFCs throughout this guide and other books in Solaris 2.5 System Administrator set.

## *How to Obtain RFCs*

The InterNIC Directory and Database Service maintains the repository of RFCs. If you have a connection to the Internet, you can retrieve online RFCs as follows:

- Through `ftp`, by sending your requests to the InterNIC directory and database server ds.internic.net. Your request should have the format:

  rfc/rfc.*rfcnum*`.txt` or `rfc/rfc.`*rfcnum*`.ps`

  where *rfcnum* represents the number of the RFC you want. For example, if you wanted to retrieve RFC 1540 in PostScript® format, you would request `rfc/rfc.1540.ps`.

- Through electronic mail, by emailing mailserv@ds.internic.net. This is an automatic server that expects the body of your request message to have the format:

  document-by-name rfc*rfcnum*

  or

  `document-by-name rfc`*rfcnum.ps*

- Through a World Wide Web browser, by specifying the URL `http://ds.internic.net/ds/dspg1intdoc.html`. The home page is

  `http://ds.internic.net`

If you need an on-line index of RFCs, send electronic mail to ds.internic.net with a message containing the request `document-by-name rfc-index`.

---

**Note –** The InterNIC information above is current as of this writing (July 1995). However, the Internet is expanding at a fast pace, and the addresses listed may no longer be current when you read this manual.

---

# *Planning Your Network* 3

This chapter describes the issues you must resolve in order to create your network in an organized, cost-effective manner. When you have resolved these issues, you can devise a plan for your network to follow as you set it up and administer it in the future.


| | |
|---|---|
| *Designing the Network* | *page 27* |
| *Designing Your IP Addressing Scheme* | *page 33* |
| *Registering Your Network* | *page 38* |
| *Naming Entities on Your Network* | *page 35* |
| *Adding Routers* | *page 40* |
| *Network Topology* | *page 40* |


If you are unfamiliar with TCP/IP fundamentals, refer to Chapter 2, “TCP/IP Protocol Suite.”

## *Designing the Network*

The first phase in the life of a network—designing the network—involves making decisions about the type of network that best suits the needs of your organization. Some of the planning decisions you make will involve network hardware; for example:

- number of host machines your network will support
- type of network media to use: Ethernet, token ring, FDDI, and so on

- network topology; that is, the physical layout and connections of the network hardware
- types of hosts the network will support: standalone, diskless, and dataless

Based on these factors, you can determine the size of your local-area network.

---

**Note –** Planning the network hardware is outside the scope of this manual. Refer to the manuals that came with your hardware for assistance.

---

## *Factors Involved in Network Planning*

After you have completed your hardware plan, you are ready to begin network planning, from the software perspective.

As part of the planning process you must:

1. Obtain a network number and, if applicable, register your network domain with the InterNIC.
2. Devise an IP addressing scheme for your hosts, after you receive your IP network number.
3. Create a list containing the IP addresses and host names of all machines to comprise your network, which you can use as you build network databases.
4. Determine which name service to use on your network: NIS, NIS+, DNS, or the network databases in the local `/etc` directory.
5. Establish administrative subdivisions, if appropriate for your network.
6. Determine if your network is large enough to require routers, and, if appropriate, create a network topology that supports them.
7. Set up subnets, if appropriate for your network.

The remainder of Chapter 3 explains how to plan your network with these factors in mind.

# Setting Up an IP Addressing Scheme

The number of machines you expect to support will affect several decisions you will need to make at this stage of setting up a network for your site. Your organization may require a small network of several dozen standalone machines located on one floor of a single building. Alternatively, you may need to set up a network with more than 1000 hosts in several buildings. This arrangement may require you to further divide your network into subdivisions called *subnets*. The size of your prospective network will affect the

- Network class you apply for
- Network number you will receive
- IP addressing scheme you then use for your network

Obtaining a network number and then establishing an IP addressing scheme is one of the most important tasks of the planning phase of network administration.

## Parts of the IP Address

Each network running TCP/IP must have a unique network number, and every machine on it must have a unique IP address. It is important to understand how IP addresses are constructed before you register your network and obtain its network number.

The IP address is a 32-bit number that uniquely identifies a network interface on a machine. An IP address is typically written in decimal digits, formatted as four 8-bit fields separated by periods. Each 8-bit field represents a byte of the IP address. This form of representing the bytes of an IP address is often referred to as the *dotted-decimal format*.

The bytes of the IP address are further classified into two parts: the network part and the host part. Figure 3-1 on page 30 shows the component parts of a typical IP address, 129.144.50.56.

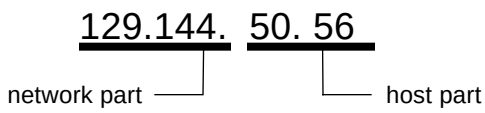



*Figure 3-1* Parts of an IP Address

### *Network Part*

This part specifies the unique number assigned to your network. It also identifies the class of network assigned. In Figure 3-1, the network part takes up two bytes of the IP address.

### *Host Part*

This is the part of the IP address that you assign to each host. It uniquely identifies this machine on your network. Note that for each host on your network, the network part of the address will be the same, but the host part must be different.

### *Subnet Number (Optional)*

Local networks with large numbers of hosts are sometimes divided into subnets. If you choose to subnet your network, you need to assign a *subnet number* for the subnet. You can maximize the efficiency of the IP address space by using some of the bits from the host number part of the IP address as a network identifier. When used as a network identifier, the specified part of the address becomes the subnet number. You create a subnet number by using a netmask, which is a bit mask that selects the network and subnet parts of an IP address. (Refer to ”Creating the Network Mask” on page 57 for full details.)

## *Network Classes*

The first step in planning for IP addressing on your network is to determine which network class is appropriate for your network. After you have done this, you can take the crucial second step: obtaining the network number from the InterNIC addressing authority.

There currently are three classes of TCP/IP networks. Each class uses the 32-bit IP address space differently, providing more or fewer bits for the network part of the address. These classes are class A, class B, and class C.

## *Class A Network Numbers*

A class A network number uses the first eight bits of the IP address as its "network part." The remaining 24 bits comprise the host part of the IP address, as illustrated in Figure 3-2 below.

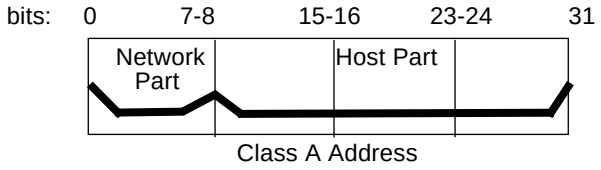



*Figure 3-2* Byte Assignment in a Class A Address

The values assigned to the first byte of class A network numbers fall within the range 0–127. Consider the IP address 75.4.10.4. The value 75 in the first byte indicates that the host is on a class A network. The remaining bytes, 4.10.4, establish the host address. The InterNIC assigns only the first byte of a class A number. Use of the remaining three bytes is left to the discretion of the owner of the network number. Only 127 class A networks can exist. Each one of these numbers can accommodate up to 16,777,214 hosts.

## *Class B Network Numbers*

A class B network number uses 16 bits for the network number and 16 bits for host numbers. The first byte of a class B network number is in the range 128–191. In the number 129.144.50.56, the first two bytes, 129.144, are assigned by the InterNIC, and comprise the network address. The last two bytes, 50.56, make up the host address, and are assigned at the discretion of the owner of the network number. Figure 3-3 on page 32 graphically illustrates a class B address.

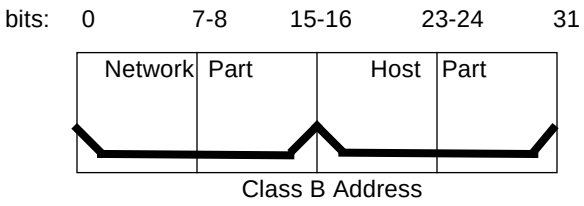



*Figure 3-3* Byte Assignment in a Class B Address

Class B typically is assigned to organizations with many hosts on their networks.

## *Class C Network Numbers*

Class C network numbers use 24 bits for the network number and 8 bits for host numbers. Class C network numbers are appropriate for networks with few hosts—the maximum being 254. A class C network number occupies the first three bytes of an IP address. Only the fourth byte is assigned at the discretion of the network owners. Figure 3-4 graphically represents the bytes in a class C address.

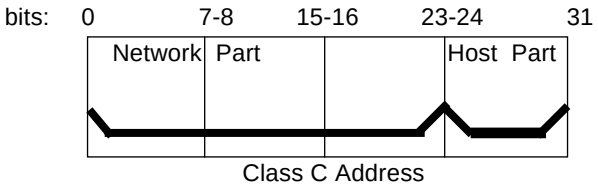



*Figure 3-4* Byte Assignment in a Class C Address

The first byte of a class C network number covers the range 192–223. The second and third each cover the range 1– 255. A typical class C address might be 192.5.2.5. The first three bytes, 192.5.2, form the network number. The final byte in this example, 5, is the host number.

## Administering Network Numbers

If your organization has been assigned more than one network number, or uses subnets, appoint a centralized authority within your organization to assign network numbers. That authority should maintain control of a pool of assigned network numbers, assigning network, subnet, and host numbers as required. To prevent problems, make sure that duplicate or random network numbers do not exist in your organization.

## Designing Your IP Addressing Scheme

After you have received your network number, you can then plan how you will assign the host parts of the IP address.

Table 3-1 shows the division of the IP address space into network and host address spaces. For each class, “range” specifies the range of decimal values for the first byte of the network number. “Network address” indicates the number of bytes of the IP address that are dedicated to the network part of the address, with each byte represented by *xxx*. “Host address” indicates the number of bytes dedicated to the host part of the address. For example, in a class A network address, the first byte is dedicated to the network, and the last three are dedicated to the host. The opposite is true for a class C network.

*Table 3-1* Division of IP Address Space

| Class | Range | Network Address | Host Address |
|---|---|---|---|
| A | 0–127 | *xxx* | *xxx.xxx.xxx* |
| B | 128–191 | *xxx.xxx* | *xxx.xxx* |
| C | 192–223 | *xxx.xxx.xxx* | *xxx* |

The numbers in the first byte of the IP address define whether the network is class A, B, or C and are always assigned by the InterNIC. The remaining three bytes have a range from 0–255. The numbers 0 and 255 are reserved; you can assign the numbers 1–254 to each byte *depending on the network number assigned to you*.

Table 3-2 shows which bytes of the IP address are assigned to you and the range of numbers within each byte that are available for you to assign to your hosts.

*Table 3-2* Range of Available Numbers

| Network Class | Byte 1 Range | Byte 2 Range | Byte 3 Range | Byte 4 Range |
|---|---|---|---|---|
| A | 0–127 | 1–254 | 1–254 | 1–254 |
| B | 128–191 | Preassigned by Internet | 1–254 | 1–254 |
| C | 192–223 | Preassigned by Internet | Preassigned by Internet | 1–254 |

## *How IP Addresses Apply to Network Interfaces*

In order to connect to the network, a computer must have at least one network interface, as explained in ”Network Interfaces” on page 5. Each network interface must have its own unique IP address. The IP address that you give to a host is assigned to its network interface, sometimes referred to as the *primary network interface*. If you add a second network interface to a machine, it must have its own unique IP number. Adding a second network interface changes the function of a machine from a host to a router, as explained in Chapter 5, “Configuring Routers.” If you add a second network interface to a host and disable routing, the host then is considered a multihomed host.

Each network interface has a device name, device driver, and associated device file in the `/devices` directory. The network interface might have a device name such as `le0` or `smc0`, device names for two commonly used Ethernet interfaces.

**Note –** This book assumes that your machines have Ethernet network interfaces. If you plan to use different network media, refer to the manuals that came with the network interface for configuration information.

## Naming Entities on Your Network

After you have received your assigned network number and given IP addresses to your hosts, the next task is to assign names to the hosts and determine how you will handle name services on your network. You will use these names when you initially set up your network and, later, for expanding your network through routers or PPP.

The TCP/IP protocols locate a machine on a network by using its IP address. However, humans find it much easier to identify a machine if it has an understandable name. Therefore, the TCP/IP protocols (and the Solaris operating system) require both the IP address and the host name to uniquely identify a machine.

From a TCP/IP perspective, a network is a set of named entities. A host is an entity with a name. A router is an entity with a name. The network is an entity with a name. A group or department in which the network is installed can also be given a name, as can a division, a region, or a company. In theory, there is virtually no limit to the hierarchy of names that can be used to identify a network and its machines. The term for these named entities is *domain*.

### Administering Host Names

Many sites let users pick host names for their machines. Servers also require at least one host name, which is associated with the IP address of its primary network interface.

As network administrator, you must ensure that each host name in your domain is unique. In other words, no two machines on your network could both have the name “fred,” although the machine “fred” might have multiple IP addresses.

When planning your network, make a list of IP addresses and their associated host names for easy access during the setup process. The list can help you verify that all host names are unique.

# 3

## Selecting a Name Service

The Solaris operating system gives you the option of using four types of name services: local files, NIS, NIS+, and DNS. Name services maintain critical information about the machines on a network, such as the host names, IP addresses, Ethernet addresses, and the like.

### Network Databases

When you install the operating system, you supply the host name and IP address of your server, clients, or standalone machine as part of the procedure. The Solaris installation program enters this information into a *network database* called the `hosts` database. The `hosts` database is one of a set of network databases that contain information necessary for TCP/IP operation on your network. These databases are read by the name service you select for your network.

Setting up the network databases is a critical part of network configuration. Therefore, you need to decide which name service to use as part of the network planning process. Moreover, the decision to use name services also affects whether or not you organize your network into an administrative domain. Chapter 4, "Configuring TCP/IP on the Network," has detailed information on the set of network databases.

### Using NIS, NIS+, or DNS for Name Service

The NIS, NIS+, or DNS name services maintain network databases on several servers on the network. *NIS+ and DNS Setup and Configuration Guide* fully describes these name services and explains how to set them up. It also explains the "namespace" and "administrative domain" concepts in complete detail. If you are changing name services from NIS to NIS+, refer to *NIS+ Transition Guide*. You should refer to these manuals to help you decide whether to use these name services on your network.

## *Using Local Files for Name Service*

If you do not implement NIS, NIS+, or DNS, the network will use *local files* to provide name service. The term “local files” refers to the series of files in the `/etc` directory that the network databases use. The procedures in this book assume you are using local files for your name service, unless otherwise indicated.

---

**Note –** If you decide to use local files as the name service for your network, you can set up another name service at a later date.

---

## *Domain Names*

Many networks organize their hosts and routers into a hierarchy of administrative domains. If you are going to use NIS, NIS+, or the DNS name services, you must select a domain name for your organization that is unique worldwide. To ensure that your domain name is unique, you should register it with the InterNIC. This is especially important if you plan to use DNS.

The domain name structure is hierarchical. A new domain typically is located below an existing, related domain. For example, the domain name for a subsidiary company could be located below the domain of the parent company. If it has no other relationship, an organization can place its domain name directly under one of the existing top-level domains.

Examples of top-level domains include:

- .com – Commercial companies (international in scope)
- .edu – Educational institutions (international in scope)
- .gov – U.S. government agencies
- .fr – France

The name that identifies your organization is one that you select, with the provision that is unique.

### Administrative Subdivisions

The question of administrative subdivisions deals with matters of size and control. The more hosts and servers you have in a network, the more complex your management task. You may wish to handle such situations by setting up additional administrative divisions in the form of more additional networks of a particular class or by dividing existing networks into subnets. The decision whether to set up administrative subdivisions for your network hinges on the following factors:

- How large is the network?

  A single network of several hundred hosts, all in the same physical location and requiring the same administrative services, could be handled by a single administrative division. On the other hand, a network of fewer machines, divided into a number of subnets and physically scattered over an extensive geographic area, would be likely to benefit from the establishment of several administrative subdivisions.

- Do users on the network have similar needs?

  For example, you may have a network that is confined to a single building and supports a relatively small number of machines. These machines are divided among a number of subnetworks, each supporting groups of users whose needs differ widely. Such a case could call for an administrative subdivision for each subnet.

  *NIS+ and FNS Administration Guide* discusses administrative subdivisions in detail.

## Registering Your Network

Before you assign IP addresses to the machines on your Solaris network, you must obtain a network number from the InterNIC. Moreover, if you plan to use administrative domains, you should register them with the InterNIC.

### InterNIC and InterNIC Registration Services

The InterNIC was created in 1993 to act as a central body where users of the Internet could go for information such as

- What the Internet's policies are

- How to access the Internet, including training services
- What resources are available to Internet users, such as anonymous FTP servers, Usenet user groups, and so on.

The InterNIC also includes the InterNIC Registration Services, the organization with which you register your TCP/IP network. The InterNIC Registration Services provide templates for obtaining a network number and for registering your domain. Two points to remember about registration are:

- The InterNIC assigns network numbers.

---

**Note –** Do not arbitrarily assign network numbers to your network, even if you do not plan to attach it to other existing TCP/IP networks.

---

  Subnet numbers are not assigned by the InterNIC. Rather, they are composed partly of the assigned network number and numbers that you define, as explained in ”What Subnetting Is” on page 56.

- You—not InterNIC—determine the domain name for your network and then register it with the InterNIC.

## *How to Contact the InterNIC*

You can reach the InterNIC Registration Services by:

- Mail
  Write to:

  Network Solutions
  Attn: InterNIC Registration Services
  505 Huntmar Park Drive
  Herndon, Virginia 22070

- Telephone
  The phone number is 1-703-742-4777. Phone service is available from 7 a.m. to 7 p.m. Eastern Standard Time.

- Electronic mail
  Send email regarding network registration to:

  Hostmaster@rs.internic.net

- Anonymous FTP or Telnet inquiries, through the Gopher and WAIS interfaces. Connect to `rs.internic.net`. (Anonymous FTP and Telnet are outside the scope of this book; however, books on these subjects are available in computer bookstores.)

## Adding Routers

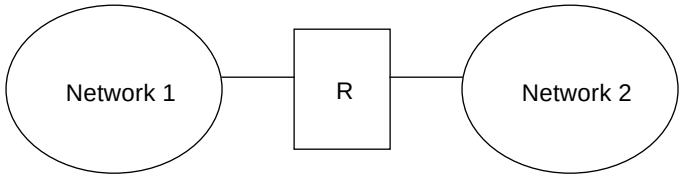
Recall that in TCP/IP, two types of entities exist on a network: hosts and routers. All networks must have hosts, while not all networks require routers. Whether you use routers should depend on the physical topology of the network. This section introduces the concepts of network topology and routing, important when you decide to add another network to your existing network environment.

### Network Topology

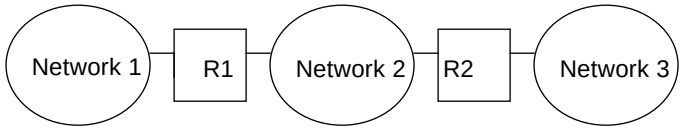
Network topology describes how networks fit together. Routers are the entities that connect networks to each other. From a TCP/IP perspective, a router is any machine that has two or more network interfaces. However, the machine cannot function as a router until properly configured, as described in Chapter 5, “Configuring Routers.”

Two or more networks can be connected together by routers to form larger internetworks. The routers must be configured to pass packets between two adjacent networks. They also should be able to pass packets to networks that lie beyond the adjacent networks.

Figure 3-5 on page 41 shows the basic parts of a network topology. The first illustration shows a simple configuration of two networks connected by a single router. The second shows a configuration of three networks, interconnected by two routers. In the first case, network 1 and network 2 are joined into a larger internetwork by router R. In the second case, router R1 connects networks 1 and 2, and router R2 connects networks 2 and 3, thus forming a network made up of networks 1, 2, and 3.




Two Networks Connected by a Router




Three Networks Connected by Two Routers

*Figure 3-5* Basic Network Topology

Routers join networks into internetworks and route packets between them based on the addresses of the destination network. As internetworks grow more complex, each router must make more and more decisions regarding where packets are to be sent.

A step up in complexity is the case shown in Figure 3-6 on page 42. Networks 1 and 3 are directly connected by a router R3. The reason for such redundancy is reliability. If network 2 goes down, router R3 still provides a route between networks 1 and 3. Any number of networks can be interconnected and communicate as long as they all adhere to the same network protocols.

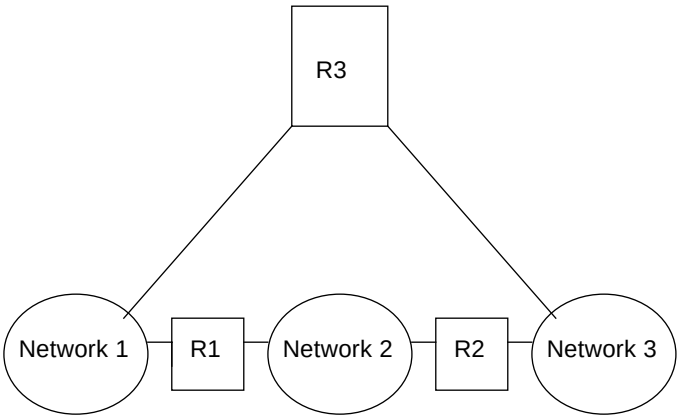



*Figure 3-6* Providing Additional Path Between Networks

## How Routers Transfer Packets

Routing decisions on a network are based on the network portion of the IP address of the recipient that is contained in the packet header. If this address includes the network number of the local network, the packet goes directly to the host with that IP address. If the network number is not the local network, the packet goes to the router on the local network

Routers maintain routing information in *routing tables*. These tables contain the IP address of the hosts and routers on the networks to which the router is connected. The tables also contain pointers to these networks. When a router gets a packet, it consults its routing table to see if it lists the destination address in the header. If the table does not contain the destination address, the router forwards the packet to another router listed in its routing table. Refer to Chapter 5, “Configuring Routers,” for detailed information on routers.

Figure 3-7 on page 43 shows a network topology with three networks connected by two routers.

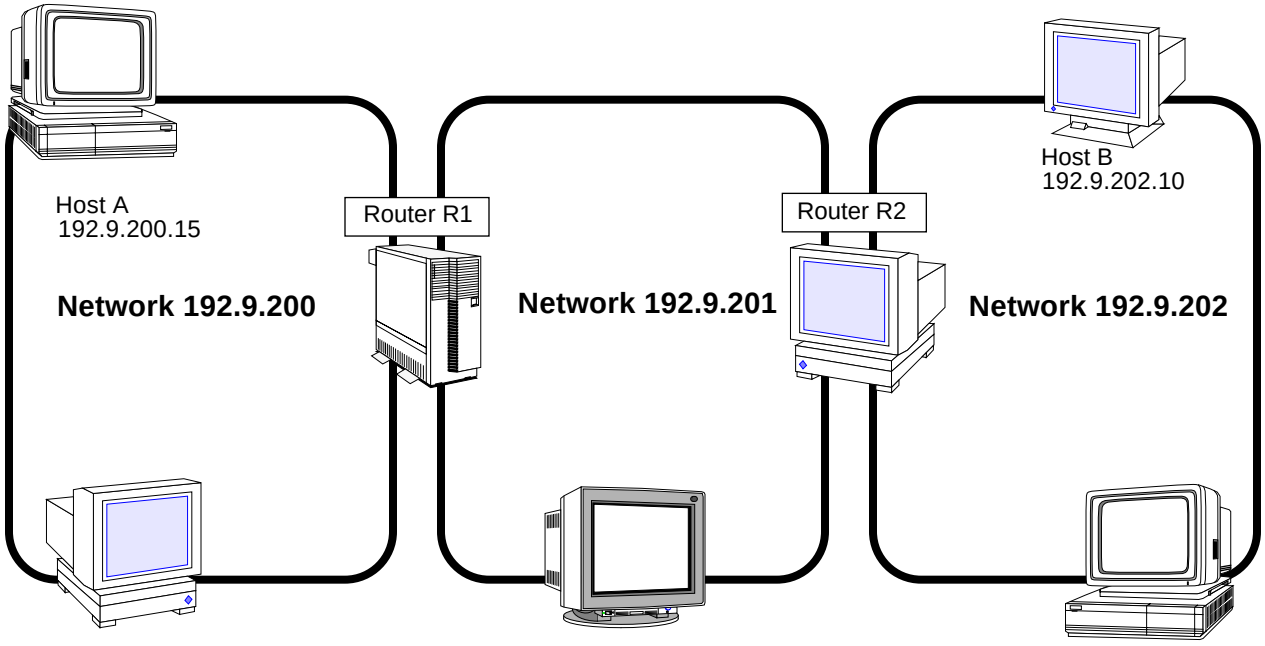



*Figure 3-7* Three Interconnected Networks

Router R1 connects networks 192.9.200 and 192.9.201. Router R2 connects networks 192.9.201 and 192.9.202. If host A on network 192.9.200 sends a message to host B on network 192.9.202, this is what happens.

1. Host A sends a packet out over network 192.9.200. The packet header contains the IP address of the recipient host B, 192.9.202.10.
2. None of the machines on network 192.9.200 has the IP address 192.9.202.10. Therefore, router R1 accepts the packet.
3. Router R1 examines its routing tables. No machine on network 192.9.201 has the address 192.9.202.10. However, the routing tables do list router R2.
4. R1 then selects R2 as the “next hop” router and sends the packet to R2.

5. Because R2 connects network 192.9.201 to 192.9.202, it has routing information for host B. Router R2 then forwards the packet to network 192.9.202, where it is accepted by host B.

# *Configuring TCP/IP on the Network* 4

The second phase of network administration involves setting up the network. This consists of assembling the hardware that makes up the physical part of the network and configuring TCP/IP. This chapter explains how to configure TCP/IP, including:

- Determining the host configuration mode for each machine on your network
- Setting up the subnet mask for your network (optional)
- Configuring TCP/IP on the machines that will run in local files mode
- Configuring a network configuration server
- Configuring TCP/IP on machines that will run in network client mode
- Editing the network databases, based on the name service you have selected for your network
- Configuring the name service switch file


| | |
|---|---|
| *Before You Configure TCP/IP* | *page 46* |
| *Determining Host Configuration Modes* | *page 47* |
| *TCP/IP Configuration Files* | *page 51* |
| *hosts Database* | *page 53* |
| *Creating the Network Mask* | *page 57* |
| *How Name Services Affect Network Databases* | *page 61* |
| *nsswitch.conf File — Specifying Which Name Service to Use* | *page 63* |
| *Network Configuration Procedures* | *page 72* |
| *Configuring Standard TCP/IP Services* | *page 77* |

# 4

## *Before You Configure TCP/IP*

Before configuring the TCP/IP software, you should have:

1. Designed the network topology, if you are the network designer. (See ”Network Topology” on page 40 for details.)
2. Obtained a network number from your Internet addressing authority. (See ”Network Classes” on page 30.)
3. Assembled the network hardware according to the topology designed and assured that the hardware is functioning. (See the hardware manuals and ”Network Topology” on page 40.)
4. Run any configuration software required by network interfaces and routers, if applicable. (See Chapter 3, “Planning Your Network,” and Chapter 5, “Configuring Routers,” for information on routers. If you have purchased network interfaces for your machines, refer to the manuals that came with them for software configuration requirements.)
5. Planned the IP addressing scheme for the network, including subnet addressing, if applicable. (See ”Designing Your IP Addressing Scheme” on page 33.)
6. Assigned IP numbers and host names to all machines involved in the network. (See ”Designing Your IP Addressing Scheme” on page 33.)
7. Determined which name service your network will use: NIS, NIS+, DNS, or local files. (See *NIS+ and FNS Administration Guide*.)
8. Selected domain names for your network, if applicable. (See *NIS+ and FNS Administration Guide*.)
9. Installed the operating system on at least one machine on the prospective network. See *SPARC: Installing Solaris Software* or *x86: Installing Solaris Software*.

# Determining Host Configuration Modes

One of the key functions you will carry out as a network administrator is configuring TCP/IP to run on hosts and routers (if applicable). You can set up these machines to obtain configuration information from two sources: files on the local machine or files located on other machines on the network. Configuration information includes:

- Host name of a machine
- IP address of the machine
- Domain name to which the machine belongs
- Default router
- Netmask in use on the machine's network

A machine that obtains TCP/IP configuration information from local files is said to be operating in *local files mode*. A machine that obtains TCP/IP configuration information from a remote machine is said to be operating in *network client mode*.

## Machines That Should Run in Local Files Mode

To run in local files mode, a machine must have local copies of the TCP/IP configuration files. These files are described in "TCP/IP Configuration Files" on page 51. The machine should have its own disk, though this is not strictly necessary.

Most servers should run in local file mode. This requirement includes:

- Network configuration servers
- NFS servers
- Name servers supplying NIS, NIS+, or DNS services
- Mail servers

Additionally, routers should run in local files mode.

Machines that exclusively function as print servers do not need to run in local files mode. Whether individual hosts should run in local files mode depends on the size of your network.

If you are running a very small network, the amount of work involved in maintaining these files on individual hosts is manageable. If your network serves hundreds of hosts, the task becomes difficult, even with the network divided into a number of administrative subdomains. Thus, for large networks,

using local files mode usually is less efficient. On the other hand, because routers and servers must be self-sufficient, they should be configured in local files mode.

### *Network Configuration Servers*

*Network configuration servers* are the machines that supply the TCP/IP configuration information to hosts configured in network client mode. These servers support three booting protocols:

- RARP – Reverse Address Resolution Protocol (RARP) maps known Ethernet addresses (48 bits) to IP addresses (32 bits), the reverse of ARP. When you run RARP on a network configuration server, this enables hosts running in network client mode to obtain their IP addresses and TCP/IP configuration files from the server. The `in.rarpd` daemon enables RARP services. Refer to the `in.rarpd`(1M) man page for complete details.
- TFTP – Trivial File Transfer Protocol (TFTP) is an application that transfers files between remote machines. The `in.tftpd` daemon carries out TFTP services, enabling file transfer between network configuration servers and their network clients.
- bootparams – The bootparams protocol supplies parameters for booting that are required by diskless clients. The `rpc.bootparamd` daemon carries out these services.

Network configuration servers may also may function as NFS file servers.

If you are going to configure any hosts as network clients, then you must also configure at least one machine on your network as a network configuration server. If your network is subnetted, then you must have at least one network configuration server for each subnet with network clients.

## *Machines That Are Network Clients*

Any host that get its configuration information from a network configuration server is said to be “operating” in network client mode. Machines configured as network clients do not require local copies of the TCP/IP configuration files.

Network client mode greatly simplifies administration of large networks. It minimizes the number of configuration tasks that must be performed on individual hosts and assures that all machines on the network adhere to the same configuration standards.

You can configure network client mode on all types of computers, from fully standalone systems to diskless and dataless machines. Although it is possible to configure routers and servers in network client mode, local files mode is a better choice for these machines. Routers and servers should be as self-sufficient as possible.

### *Diskless Booting*

Setting up systems for diskless booting is described in *System Administration Guide, Volume II*.

## *Mixed Configurations*

Due to the flexibility of the system, configurations are not limited to either an all local hosts mode or an all network client mode. The configuration of routers and servers typifies this, in that routers and servers should always be configured in local mode. For hosts, you can use any combination of local and network client mode you wish.

## *Sample Network*

Figure 4-1 on page 50 shows the hosts of a fictional network with the network number 192.9.200. The network includes one network configuration server, the machine `sahara`. It serves the diskless client `ahaggar`. Machines `tenere` and `nubian` have their own disks and run in local files mode. Machine `faiyum` also has a disk but operates in network client mode.

Finally, the machine `timbuktu` is configured as a router. It includes two network interfaces, one named `timbuktu` on network 192.9.200 and one named `timbuktu-201` on network 192.9.201. Both networks are in the organizational domain deserts.worldwide.com. The domain uses local files as its name service.

Most examples in this chapter use the network shown in Figure 4-1 on page 50 as their basis.

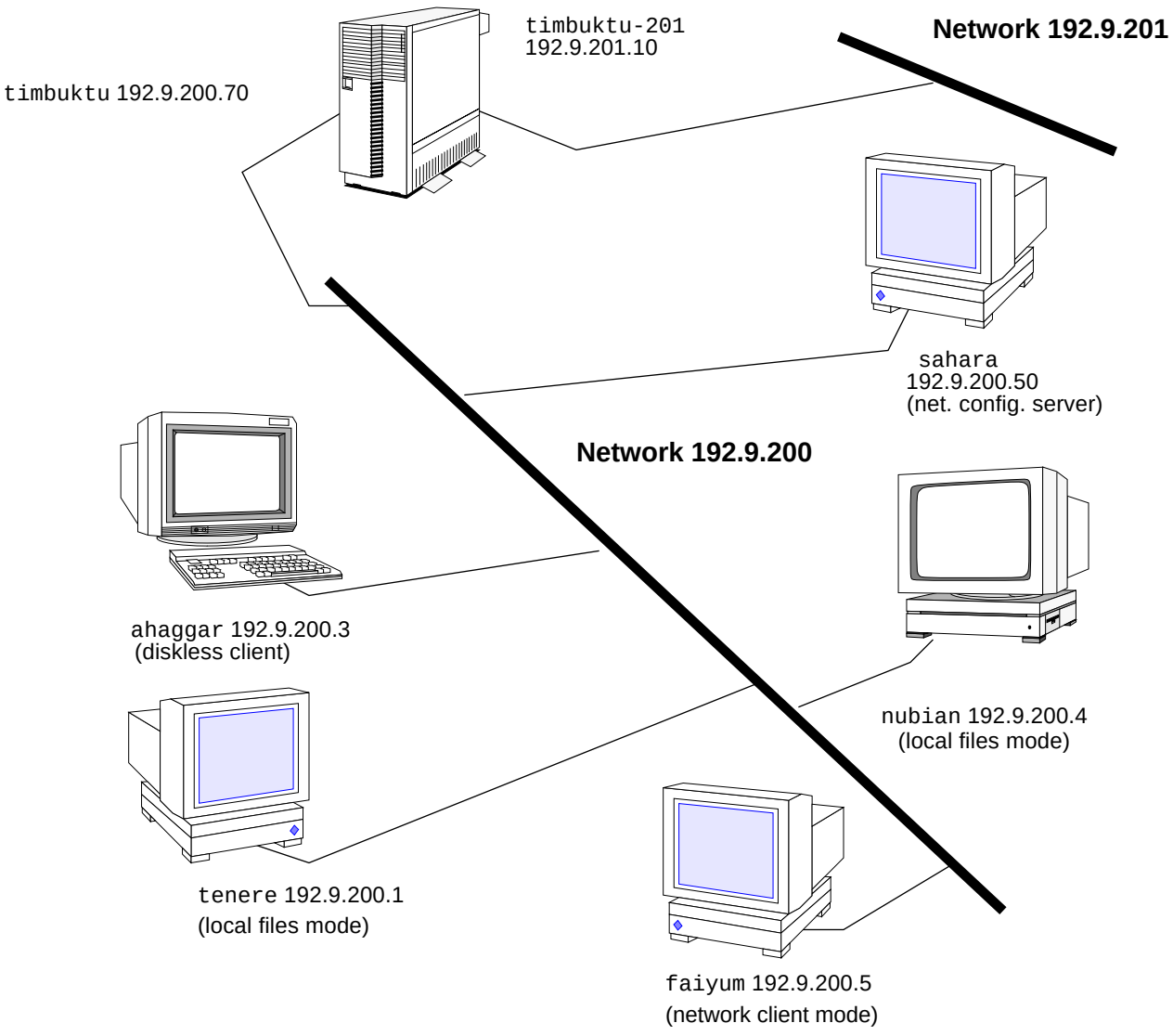



*Figure 4-1* Hosts in a Sample Network

# TCP/IP Configuration Files

Each machine on the network gets its TCP/IP configuration information from the following TCP/IP configuration files and network databases:

- `/etc/hostname.`*interface* file
- `/etc/nodename` file
- `/etc/defaultdomain` file
- `/etc/defaultrouter` file (optional)
- `hosts` database
- `netmasks` database (optional)

The Solaris installation program creates these files as part of the installation process. You can also edit the files manually, as explained in "TCP/IP Configuration Files". The `hosts` and `netmasks` databases are two of the network databases read by the name services available on Solaris networks. "Network Databases and nsswitch.conf File" on page 60 describes the concept of network databases in detail.

## `/etc/hostname.`*interface* File

This file defines the network interfaces on the local host. At least one `/etc/hostname.`*interface* file should exist on the local machine. The Solaris installation program creates this file for you. In the file name, *interface* is replaced by the device name of the primary network interface.

The file contains only one entry: the host name or IP address associated with the network interface. For example, suppose `smc0` is the primary network interface for a machine called `ahaggar`. Its `/etc/hostname.`*interface* file would have the name `/etc/hostname.smc0`; the file would contain the entry `ahaggar`.

### For Multiple Network Interfaces

If a machine contains more than one network interface, you must create additional `/etc/hostname.`*interface* files for the additional network interfaces. You must create these files with a text editor; the Solaris installation program will not create them for you.

For example, consider the machine `timbuktu` shown in Figure 4-1 on page 50. It has two network interfaces and functions as a router. The primary network interface `le0` is connected to network 192.9.200. Its IP address is 192.9.200.70, andits host name is `timbuktu`.The Solaris installation program creates the file `/etc/hostname.le0` for the primary network interface and enters the host name `timbuktu` in the file.

The second network interface is `le1`; it is connected to network 192.9.201. Although this interface is physically installed on machine `timbuktu`, it must have a separate IP address. Therefore, you have to manually create the `/etc/hostname.le1` file for this interface; the entry in the file would be the router‘s name `timbuktu-201`.

## `/etc/nodename` *File*

This file should contain one entry: the host name of the local machine. For example, on machine `timbuktu`, the file `/etc/nodename` would contain the entry `timbuktu`.

## `/etc/defaultdomain` *File*

This file should contain one entry, the fully qualified domain name of the administrative domain to which the local host’s network belongs. You can supply this name to the Solaris installation program or edit the file at a later date.

In Figure 4-1 on page 50, the networks are part of the domain deserts.worldwide, which was classified as a .com domain. Therefore, `/etc/defaultdomain` should contain the entry `deserts.worldwide.com`. For more information on network domains, refer to *NIS+ and FNS Administration Guide.*

## `/etc/defaultrouter` *File*

This file should contain an entry for each router directly connected to the network. The entry should be the name for the network interface that functions as a router between networks.

In Figure 4-1 on page 50, the network interface `le1` connects machine `timbuktu` with network 192.9.201. This interface has the unique name `timbuktu-201`. Thus, the machines on network 192.9.200 that are configured in local files mode have the name `timbuktu-201` as the entry in `/etc/defaultrouter`.

## `hosts` *Database*

The `hosts` database contains the IP addresses and host names of machines on your network. If you use the NIS, NIS+, or DNS name services, the `hosts` database is maintained in a database designated for host information. For example, on a network running NIS+, the `hosts` database is maintained in the host table.

If you use local files for name service, the `hosts` database is maintained in the `/etc/inet/hosts` file. This file contains the host names and IP addresses of the primary network interface, other network interfaces attached to the machine, and any other network addresses that the machine must know about.

---

**Note –** For compatibility with BSD-based operating systems, the file `/etc/hosts` is a symbolic link to `/etc/inet/hosts`.

---

### `/etc/inet/hosts` *File Format*

The `/etc/inet/hosts` file uses this basic syntax: (Refer to the `hosts`(4) man page for complete syntax information.)

*IP-address hostname [nicknames] [#comment]*

***IP-address***
contains the IP address for each interface that the local host must know about.

***hostname***
contains the host name assigned to the machine at setup, plus the host names assigned to additional network interfaces that the local host must know about.

***[nickname]***
is an optional field containing a nickname for the host.

***[# comment]***
is an optional field where you can include a comment.

## *Initial* `/etc/inet/hosts` *File*

When you run the Solaris installation program on a machine, it sets up the initial `/etc/inet/hosts` file. This file contains the minimum entries that the local host requires: its loopback address, its IP address, and its host name.

For example, the Solaris installation program might create the following `/etc/inet/hosts` file for machine `ahaggar` shown in Figure 4-1 on page 50:

*Code Example 4-1* `/etc/inet/hosts` File for Machine `ahaggar`

```
127.0.0.1       localhost         loghost    #loopback address
192.9.200.3     ahaggar                      #host name
```

### *Loopback Address*

In Code Example 4-1, the IP address 127.0.0.1 is the *loopback address*, the reserved network interface used by the local machine to allow interprocess communication so that it sends packets to itself. The `ifconfig` command uses the loopback address for configuration and testing, as explained in ”ifconfig Command” on page 92. Every machine on a TCP/IP network must use the IP address 127.0.0.1 for the local host.

### *Host Name*

The IP address 192.9.200.3 and the name `ahaggar` are the address and host name of the local machine. They are assigned to the machine’s primary network interface.

### *Multiple Network Interfaces*

Some machines have more than one network interface, either because they are routers or multihomed hosts. Each additional network interface attached to the machine requires its own IP address and associated name. When you configure a router or multihomed host, you must manually add this information to the router’s `/etc/inet/hosts` file. (See Chapter 5, “Configuring Routers,” for more information on setting up routers and multihomed hosts.)

Here is the `/etc/inet/hosts` file for machine `timbuktu` shown in Figure 4-1 on page 50.

*Code Example 4-2* `/etc/inet/hosts` File for Machine `timbuktu`

```
 127.0.0.1       localhost     loghost
192.9.200.70     timbuktu      #This is the local host name
192.9.201.10     timbuktu-201  #Interface to network 192.9.201
```

With these two interfaces, `timbuktu` connects networks 192.9.200 and 192.9.201 as a router.

## *How Name Services Affect the* `hosts` *Database*

The NIS, NIS+, and DNS name services maintain host names and addresses on one or more servers. These servers maintain `hosts` databases containing information for every host and router (if applicable) on the servers' network. Refer to *NIS+ and FNS Administration Guide* for more information about these services.

### *When Local Files Provide Name Service*

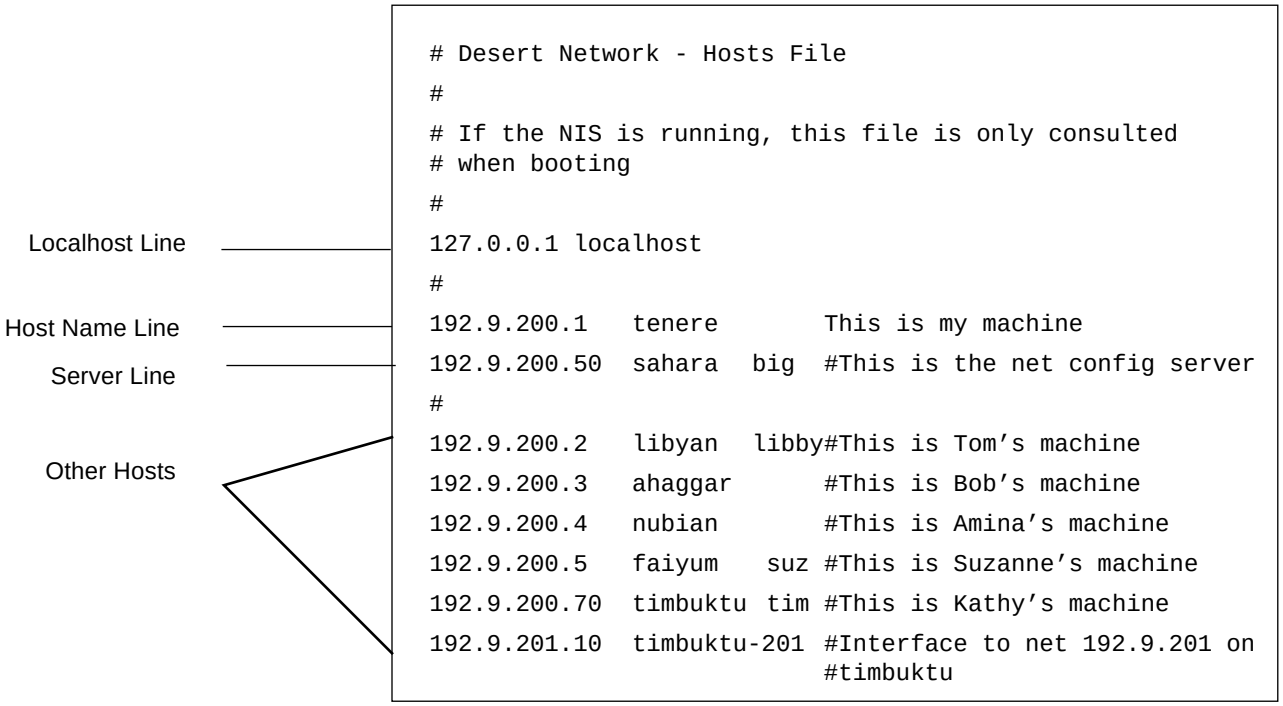
On a network using local files for name service, machines running in local files mode consult their individual `/etc/inet/hosts` files for IP addresses and host names of other machines on the network. Therefore, their `/etc/inet/hosts` files must contain the:

- Loopback address
- IP address and host name of the local machine (primary network interface)
- IP address and host name of additional network interfaces attached to this machine, if applicable
- IP addresses and host names of all hosts on the local network
- IP addresses and host names of any routers this machine must know about, if applicable
- IP address of any machine your machine wants to refer to by its host name

Code Example 4-3 on page 56 shows the `/etc/inet/hosts` file for machine `tenere`, a machine that runs in local files mode. Notice that the file contains the IP addresses and host names for every machine on the 192.9.200 network. It also contains the IP address and interface name `timbuktu-201`, which connects the 192.9.200 network to the 192.9.201 network.

A machine configured as a network client uses the local `/etc/inet/hosts` file for its loopback address and IP address.

*Code Example 4-3* `/etc/inet/hosts` File for Machine Running in Local Files Mode




```
# Desert Network - Hosts File
#
# If the NIS is running, this file is only consulted
# when booting
#
127.0.0.1 localhost
#
192.9.200.1   tenere        This is my machine
192.9.200.50  sahara   big  #This is the net config server
#
192.9.200.2   libyan   libby#This is Tom's machine
192.9.200.3   ahaggar       #This is Bob's machine
192.9.200.4   nubian        #This is Amina's machine
192.9.200.5   faiyum    suz #This is Suzanne's machine
192.9.200.70  timbuktu  tim #This is Kathy's machine
192.9.201.10  timbuktu-201  #Interface to net 192.9.201 on
                            #timbuktu
```

## `netmasks` *Database*

You need to edit the `netmasks` database as part of network configuration *only* if you have set up subnetting on your network. The `netmasks` database consists of a list of networks and their associated subnet masks.

**Note –** When you create subnets, each new network must be a separate physical network. You cannot apply subnetting to a single physical network.

### *What Subnetting Is*

Subnetting is a method for getting the most out of the limited 32-bit IP addressing space and reducing the size of the routing tables in a large internetwork. With any address class, subnetting provides a means of

allocating a part of the host address space to network addresses, which lets you have more networks. The part of the host address space allocated to new network addresses is known as the *subnet* number.

In addition to making more efficient use of the IP address space, subnetting has several administrative benefits. Routing can get very complicated as the number of networks grows. A small organization, for example, might give each local network a class C number. As the organization grows, administering a number of different network numbers could become complicated. A better idea is to allocate a few class B network numbers to each major division in an organization. For instance, you could allocate one to Engineering, one to Operations, and so on. Then, you could divide each class B network into additional networks, using the additional network numbers gained by subnetting. This can also reduce the amount of routing information that must be communicated among routers.

## *Creating the Network Mask*

As part of the subnetting process, you will need to select a network-wide netmask. The netmask determines how many and which bits in the host address space will represent the subnet number and how many and which will represent the host number. Recall that the complete IP address consists of 32 bits. Depending on the address class, as many as 24 bits and as few as 8 bits can be available for representing the host address space. The netmask is specified in the `netmasks` database.

If you plan to use subnets, you must determine your netmask before you configure TCP/IP. You then need to carry out the procedures in ”How to Add a Subnet to a Network” on page 60. If you plan to install the operating system as part of network configuration, the Solaris installation program will request the netmask for your network.

As described in ”Parts of the IP Address” on page 29, 32-bit IP addresses consist of a network part and a host part. The 32 bits are divided into 4 bytes. Each byte is assigned either to the network number or the host number, depending on the network class.

For example, in a class B IP address, the 2 left–hand bytes are assigned to the network number, and the 2 right–hand bytes are assigned to the host number. In the class B IP address 129.144.41.10, you can assign the 2 right–hand bytes to hosts.

If you are going to implement subnetting, you will need to use some of the bits in the bytes assigned to the host number to apply to subnet addresses. For example, a 16-bit host address space provides addressing for 65,534 hosts. If you apply the third byte to subnet addresses and the fourth to host addresses, you can address up to 254 networks, with up to 254 hosts on each.

Which bits in the host address bytes will be applied to subnet addresses and which to host addresses is determined by a subnet mask. Subnet masks are used to select bits from either byte for use as subnet addresses. Although netmask bits must be contiguous, they need not align on byte boundaries.

The netmask can be applied to an IP address using the bitwise logical AND operator. This operation selects out the network number and subnet number positions of the address.

It is easiest to explain netmasks in terms of their binary representation. You can use a calculator for binary-to-decimal conversion. The following examples show both the decimal and binary forms of the netmask.

If a netmask 255.255.255.0 is applied to the IP address 129.144.41.101, the result is the IP address of 129.144.41.0.

129.144.41.101 & 255.255.255.0 = 129.144.41.0

In binary form, the operation is:

11111111.11111111.11111111.00000000 (netmask)

AND

10000001.10010000.00101001.01100101 (IP address)

Now the system will look for a network number of 129.144.41 instead of a network number of 129.144. If you have a network with the number 129.144.41, that is what the system will look for and find. Since you can assign up to 254 values to the third byte of the IP address space, subnetting lets you create address space for 254 networks where previously there was room for only one.

If you want to provide address space for only two additional networks, you could use a subnet mask of:

255.255.192.0

This netmask provides a result of:

11111111.11111111.1100000.00000000

This still leaves 14 bits available for host addresses. Since all 0s and 1s are reserved, at least two bits must be reserved for the host number.

## *Editing the* `/etc/inet/netmasks` *File*

If your network runs NIS or NIS+, the servers for these name services maintain `netmasks` databases. For networks that use local files for name service, this information is maintained in the `/etc/inet/netmasks` file.

---

**Note –** For compatibility with BSD-based operating systems, the file `/etc/netmasks` is a symbolic link to `/etc/inet/netmasks`.

---

Here is a sample `/etc/inet/netmasks` file for a class B network.

*Code Example 4-4* `/etc/inet/netmasks` File for a Class B Network

```
## The netmasks file associates Internet Protocol (IP) address
# masks with IP network numbers.
#
#    network-numbernetmask
#
# Both the network-number and the netmasks are specified in
# “decimal dot” notation, e.g:
#
#        128.32.0.0   255.255.255.0
         129.144.0.0  255.255.255.0
```

If the file does not exist, create it. Use the following syntax:

```
network-number          netmask-number
```

Refer to the `netmasks`(4) man page for complete details.

When creating netmask numbers, type the network number assigned by the InterNIC (not the subnet number) and netmask number in `/etc/inet/netmasks`. Each subnet mask should be on a separate line.

For example:

```
128.78.0.0    255.255.248.0
```

You also can type symbolic names for network numbers in the `/etc/inet/hosts` file. You can then use these network names instead of the network numbers as parameters to commands.

### ▼ How to Add a Subnet to a Network

If you are changing from a network that does not use subnets to one that is subnetted, perform the following steps:

1. **Decide on the new subnet topology, including considerations for routers and locations of hosts on the subnets.**
2. **Assign all subnet and host addresses.**
3. **Modify the `/etc/inet/netmasks` file, if you are manually configuring TCP/IP, or supply the netmask to the Solaris installation program.**
4. **Modify the `/etc/inet/hosts` files on all hosts to reflect the new host addresses.**
5. **Reboot all machines.**

## *Network Databases and* `nsswitch.conf` *File*

The network databases are files that provide information needed to configure the network. The network databases are

- `hosts`
- `netmasks`
- `ethers`
- `bootparams`
- `protocols`
- `services`
- `networks`

As part of the configuration process, you edit the `hosts` database and the `netmasks` database, if your network is subnetted. Two network databases, `bootparams` and `ethers`, are used to configure machines as network clients. The remaining databases are used by the operating system and seldom require editing.

Although it is not a network database, the `nsswitch.conf` file needs to be configured along with the relevant network databases. `nsswitch.conf` specifies which name service to use for a particular machine: NIS, NIS+, DNS, or local files.

## *How Name Services Affect Network Databases*

The form a network database takes depends on the type of name service you select for your network. For example, the `hosts` database contains, at minimum, the host name and IP address of the local machine and any network interfaces directly connected to the local machine. However, the `hosts` database could contain other IP addresses and host names, depending on the type of name service in use on your network.

The network databases are used as follows:

- Networks that use local files for their name service use files in the `/etc/inet` and `/etc` directories
- NIS+ uses databases called NIS+ tables
- NIS uses databases called NIS maps
- DNS uses records with host information

---

**Note –** DNS boot and data files do not correspond directly to the network databases.

---

Figure 4-2 on page 62 shows the forms of the `hosts` database used by these name services:

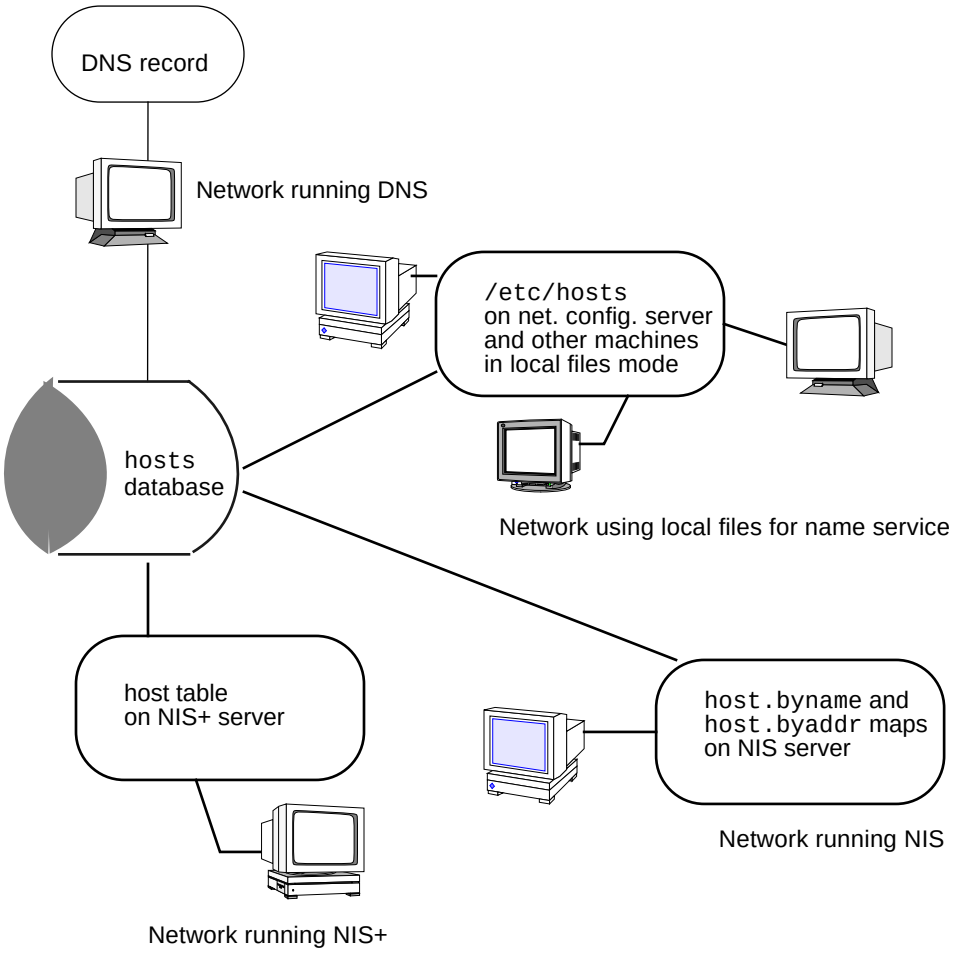



*Figure 4-2* Forms of the `hosts` Database Used by Name Services

Table 4-1 on page 63 lists the network databases and how they are used by local files, NIS+, and NIS.

*Table 4-1* Network Databases and Corresponding Name Service Files

| Network Database | Local Files | NIS+ Tables | NIS Maps |
|---|---|---|---|
| `hosts` | `/etc/inet/hosts` | hosts.ord_dir | `hosts.byaddr`<br>`hosts.byname` |
| `netmasks` | `/etc/inet/netmasks` | netmasks.ord_dir | `netmasks.byaddr` |
| `ethers` | `/etc/ethers` | ethers.ord_dir | `ethers.byname`<br>`ethers.byaddr` |
| `bootparams` | `/etc/bootparams` | bootparams.ord_dir | `bootparams` |
| `protocols` | `/etc/inet/protocols` | protocols.ord_dir | `protocols.byname`<br>`protocols.bynumber` |
| `services` | `/etc/inet/services` | services.ord_dir | `services.byname` |
| `networks` | `/etc/inet/networks` | networks.ord_dir | `networks.byaddr`<br>`networks.byname` |

This book discusses network databases as they are used by networks using local files for name services. Information regarding the `hosts` database is in ”hosts Database” on page 53; information regarding the `netmasks` database is in ”netmasks Database” on page 56. Refer to *NIS+ and FNS Administration Guide* for information on network databases correspondences in NIS, DNS, and NIS+.

## `nsswitch.conf` *File — Specifying Which Name Service to Use*

The `/etc/nsswitch.conf` file defines the search order of the network databases. The Solaris installation program creates a default `/etc/nsswitch.conf` file for the local machine, based on the name service you indicate during the installation process. If you selected the “None” option, indicating local files for name service, the resulting `nsswitch.conf file` will resemble Code Example 4-5 on page 64.

*Code Example 4-5* `nsswitch.conf` for Networks Using Files for Name Service

```
# /etc/nsswitch.files:
#
# An example file that could be copied over to /etc/nsswitch.conf;
# it does not use any naming service.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file contains "switch.so" as a
# nametoaddr library for "inet" transports.

passwd:     files
group:      files
hosts:      files
networks:   files
protocols:  files
rpc:        files
ethers:     files
netmasks:   files
bootparams: files
publickey:  files
# At present there isn't a 'files' backend for netgroup; the
# system will figure it out pretty quickly,
# and won't use netgroups at all.
netgroup:   files
automount:  files
aliases:    files
services:   files
sendmailvars:   files
```

The `nsswitch.conf`(4) man page describes the file in detail. Its basic syntax is

*database name-service-to-search*

The *database* field can list one of many types of databases searched by the operating system. For example, it could indicate a database affecting users, such as `passwd` or `aliases`, or a network database. The parameter *name-service-to-search* can have the values `files`, `nis`, or `nis+` for the network databases. (The `hosts` database can also have `dns` as a name service to search.) You can also list more than one name service, such as `nis+` and `files`.

In  on page 64, the only search option indicated is `files`. Therefore, the local machine gets security and automounting information, in addition to network database information, from files located in its `/etc` and `/etc/inet` directories.

## *Changing* `nsswitch.conf`

The `/etc` directory contains the `nsswitch.conf` file created by the Solaris installation program. It also contains template files for the following name services:

- `nsswitch.files`
- `nsswitch.nis`
- `nsswitch.nis+`

If you want to change from one name service to another, you can copy the appropriate template to `nsswitch.conf`. You can also selectively edit the `nsswitch.conf` file, and change the default name service to search for individual databases.

For example, on a network running NIS, you may have to change the `nsswitch.conf` file on diskless clients. The search path for the `bootparams` and `ethers` databases must list `files` as the first option, and then `nis`. Code Example 4-6 on page 66 shows the correct search paths.

*Code Example 4-6* `nsswitch.conf` for a Diskless Client on a Network Running NIS

```
## /etc/nsswitch.conf:
#
.
.
passwd:     files nis
group:      files nis

# consult /etc "files" only if nis is down.
hosts:      nis [NOTFOUND=return] files
networks:   nis [NOTFOUND=return] files
protocols:  nis [NOTFOUND=return] files
rpc:        nis [NOTFOUND=return] files
ethers:     files [NOTFOUND=return] nis
netmasks:   nis [NOTFOUND=return] files
bootparams: files [NOTFOUND=return] nis
publickey:  nis [NOTFOUND=return] files

netgroup:   nis

automount:  files nis
aliases:    files nis

# for efficient getservbyname() avoid nis
services:   files nis
sendmailvars:   files
```

For complete details on the name service switch, refer to *NIS+ and FNS Administration Guide.*

## bootparams *Database*

The bootparams database contains information used by diskless clients and machines configured to boot in the network client mode. You will have to edit it if your network will have network clients. (See ”Configuring Network Clients” on page 75 for procedures.) The database is built from information entered into the /etc/bootparams file.

The bootparams(4) man page contains complete syntax for this database. Its basic syntax is

*machine-name file-key-server-name:pathname*

For each diskless or network client machine, the entry may contain the following information: the name of the client, a list of keys, the names of servers, and path names.

The first item of each entry is the name of the client machine. Next is a list of keys, names of servers, and path names, separated by tab characters. All items but the first are optional. The database can contain a wildcard entry that will be matched by all clients. Here is an example:

*Code Example 4-7* bootparams Database

```
myclient   root=myserver : /nfsroot/myclient  \
swap=myserver : /nfsswap//myclient \
dump=myserver : /nfsdump/myclient
```

In this example the term dump=: tells diskless hosts not to look for a dump file.

### *Wildcard Entry for* bootparams

In most cases, you will want to use the wildcard entry when editing the bootparams database to support diskless clients. This entry is:

```
*  root=server:/path dump=:
```

The asterisk (*) wildcard indicates that this entry applies to all clients not specifically named within the `bootparams` database.

## `ethers` *Database*

The `ethers` database is built from information entered into the `/etc/ethers` file. It associates host names to their Ethernet addresses. You need to create an `ethers` database only if you are running the RARP daemon; that is, if you are configuring network clients or diskless machines.

RARP uses the file to map Ethernet addresses to IP addresses. If you are running the RARP daemon `in.rarpd`, you need to set up the `ethers` file and maintain it on all hosts running the daemon to reflect changes to the network.

The `ethers`(4) man page contains complete syntax information for this database. Its basic format is:

*Ethernet-address hostname #comment*

***Ethernet-address***
is the Ethernet address of the host.

***hostname***
is the official name of the host.

***#comment***
is any kind of note you want to append to an entry in the file.

The equipment manufacturer provides the Ethernet address. If a machine does not display the Ethernet address when you power up, see your hardware manuals for assistance.

When adding entries to the `ethers` database, make sure that host names correspond to the primary names in the `hosts` database, not to the nicknames.

*Code Example 4-8* Entries in the `ethers` Database

```
8:0:20:1:40:16    fayoum
8:0:20:1:40:15    nubian
8:0:20:1:40:7     sahara          # This is a comment
8:0:20:1:40:14    tenere
```

## *Other Network Databases*

The remaining network databases seldom need to be edited.

### `networks` *database*

The `networks` database associates network names with network numbers, enabling some applications to use and display names rather than numbers. The `networks` database is based on information in the `/etc/inet/networks` file. It contains the names of all networks to which your network connects via routers.

The Solaris installation program sets up the initial `networks` database. The only time you need to update it is when you add a new network to your existing network topology.

The `networks`(4) man page contains full syntax information for `/etc/inet/networks`. Here is its basic format:

*network-name network-number nickname(s) #comment*

***network-name***
is the official name for the network.

***network-number***
is the number assigned by the InterNIC.

***nickname***
is any other name by which the network is known.

***#comment***
is any kind of note you want to append to an entry in the file.

It is particularly important that you maintain the `networks` file. The `netstat` program uses the information in this database to produce status tables.

Here is a sample `/etc/networks` file:

*Code Example 4-9* `/etc/networks` File

```
#ident"@(#)networks1.492/07/14 SMI"/* SVr4.0 1.1*/
#
# The networks file associates Internet Protocol (IP) network
numbers with network names. The format of this file is:
#
#   network-name  network-number  nicnames . . .

# The loopback network is used only for intra-machine
communication
#
loopback    127

# Internet networks
#
arpanet 10      arpa# Historical
ucb-ether 46ucbether
#
# local networks
eng           193.9.0   #engineering
acc           193.9.1   #accounting
prog          193.9.2   #programming
```

## `protocols` *Database*

The `protocols` database lists the TCP/IP protocols installed on your system and their numbers; the Solaris installation program automatically creates it. It is rare when this file requires administrative handling.

The `protocols` database contains the names of the TCP/IP protocols installed on the system. Its syntax is completely described in the `protocols`(4) man page. Here is an example of the `/etc/inet/protocols` file:

*Code Example 4-10* `/etc/inet/protocols` File

```
#
# Internet (IP) protocols
#
ip  0   IP  # internet protocol, pseudo protocol number
icmp 1  ICMP# internet control message protocol
tcp 6   TCP # transmission control protocol
udp 17  UDP # user datagram protocol
```

## `services` *Database*

The `services` database lists the names of TCP and UDP services and their well known port numbers; it is used by programs that call network services. The Solaris installation automatically creates the `services` database; it generally requires no administrative handling.

The `services`(4) man page contains complete syntax information. Here is an excerpt from a typical `/etc/inet/services` file:

*Code Example 4-11* `/etc/inet/services` File

```
#
# Network services
#
echo      7/udp
echo      7/tcp
discard   9/udp     sink null
discard   9/tcp     sink null
systat    11/tcp
daytime   13/udp
daytime   13/tcp
netstat   15/tcp
ftp-data  20/tcp
ftp       21/tcp
telnet    23/tcp
time      37/tcp    timeserver
time      37/udp    timeserver
name      42/udp    nameserver
whois     43/tcp    nickname
```

# 4

# *Network Configuration Procedures*

Network software installation takes place along with the installation of the operating system software. At that time, certain IP configuration parameters must be stored in appropriate files so they can be read at boot time.

The procedure is simply a matter of creating or editing the network-configuration files. How configuration information is made available to a machine's kernel depends on whether these files are stored locally (local files mode) or acquired from the network configuration server (network client mode).

Parameters supplied during network configuration are:

- IP address of each network interface on every machine.
- Host names of each machine on the network. You can type the host name in a local file or a name service database.
- NIS, NIS+, or DNS domain name in which the machine resides, if applicable.
- Default router addresses. You supply this only if you have a simple network topology with only one router attached to each network, or your routers don't run routing protocols such as the Router Discovery Server Protocol (RDISC) or the Router Information Protocol (RIP). (See Chapter 5, "Configuring Routers," for more information about these protocols.)
- Subnet mask (required only for networks with subnets).

This chapter contains complete information on creating and editing local configuration files. See *NIS+ and FNS Administration Guide* for information on working with name service databases.

## ▼ How to Configure a Host for Local Files Mode

Use this procedure for configuring TCP/IP on a machine that will run in local files mode.

1. **Become superuser and change to the `/etc` directory.**
2. **Type the host name of the machine in the file `/etc/nodename`.**
   For example, if the name of the host is `tenere`, type `tenere` in the file.

3. **Create a file named** `/etc/hostname.`*interface* **for each network interface.**
   (The Solaris installation program automatically creates this file for the primary network interface.)

   Refer to "/etc/hostname.interface File" on page 51 for complete details.

4. **Type either the interface IP address or the interface name in each** `/etc/hostname.`*interface* **file.**
   For example, create a file named `hostname.ie1`, and type either the IP address of the host's interface or the host's name.

5. **Edit the** `/etc/inet/hosts` **file to add:**

---

**Note –** The Solaris installation program creates the default `/etc/inet/hosts` for the local machine. If the file does not exist, create it as shown in "hosts Database" on page 53.

---

   a. **IP addresses that you have assigned to any additional network interfaces in the local machine, along with the corresponding host name for each interface.**
      The Solaris installation program will already have created entries for the primary network interface and loopback address.

   b. **IP address or addresses of the file server, if the** `/usr` **file system is NFS mounted.**

6. **Type the host's fully qualified domain name in the** `/etc/defaultdomain` **file.**

   For example, suppose host `tenere` was part of the domain deserts.worldwide.com. Therefore, you would type `deserts.worldwide.com` in `/etc/defaultdomain`. See "/etc/defaultdomain File" on page 52 for more information.

7. **Type the router's name in** `/etc/defaultrouter`**.**
   See "/etc/defaultrouter File" on page 52 for information about this file.

8. **Type the name of the default router and its IP addresses in** `/etc/inet/hosts`**.**

   Additional routing options are available. Refer to the discussion on routing options in "How to Configure Hosts for Network Client Mode" on page 75. You can apply these options to a local files mode configuration.

9. **If your network is subnetted, type the network number and the netmask in the file `/etc/inet/netmasks`.**

   If you have set up a NIS or NIS+ server, you can type `netmask` information in the appropriate database on the server as long as server and clients are on the same network.

10. **Reboot each machine on the network.**

## *Setting Up a Network Configuration Server*

If you plan to configure certain hosts as network clients, you must configure at least one machine on your network as a network configuration server. (Refer to "Network Configuration Servers" on page 48 for an introduction.)

Setting up a network configuration server involves:

1. Turning on the network configuration daemons:

- `in.tftpd`
- `in.rarpd`
- `rpc.bootparamd`

2. Editing and maintaining the network configuration files on the configuration server.

"How to Set Up a Network Configuration Server" assumes that you have already set up the network configuration server for local files mode.

### ▼ How to Set Up a Network Configuration Server

1. **Become superuser and change to the root directory of the prospective network configuration server.**

2. **Turn on the `in.tftpd` daemon by creating the directory `/tftpboot`:**

```
# mkdir /tftpboot
```

   This configures the machine as a TFTP, bootparams, and RARP server.

3. **Edit the `hosts` database, and add the host names and IP addresses for every client on the network.**

4. **Edit the `ethers` database, and create entries for every host on the network to run in network client mode.**

5. **Edit the `bootparams` database.**
   See ”bootparams Database” on page 67. Use the wildcard entry or create an entry for every host that will run in network client mode.

6. **Reboot the server.**

Information for setting up diskless clients, install servers, and boot servers can be found in *SPARC: Installing Solaris Software* and *x86: Installing Solaris Software*.

## *Configuring Network Clients*

Network clients receive their configuration information from network configuration servers. Therefore, before you configure a host as a network client you must ensure that at least one network configuration server is set up for the network.

### ▼ How to Configure Hosts for Network Client Mode

Do the following on each host to be configured in network client mode:

1. **Become superuser.**

2. **Check the directory for the existence of an `/etc/nodename` file. If one exists, delete it.**
   Eliminating `/etc/nodename` causes the system to use the `hostconfig` program to obtain the host name, domain name, and router addresses from the network configuration server. See ”Network Configuration Procedures” on page 72.

3. **Create the file `/etc/hostname.`*interface*, if it does not exist.**
   Make sure that the file is empty. An empty `/etc/hostname.`*interface* file causes the system to acquire the IP address from the network configuration server.

4. **Ensure that the `/etc/inet/hosts` file contains only the host name and IP address of the loopback network interface.**
   (See ”Loopback Address” on page 54). The file should not contain the IP address and host name for the local machine (primary network interface).

*EXCEPTION:* For a diskless client (a machine with an NFS-mounted root file system), type the name and IP address of the server that provides the client's root file system (usually, but not always, the network configuration server).

5. **Check for the existence of an `/etc/defaultdomain` file. If one exists, delete it.**
   The `hostconfig` program will set the domain name automatically. If you wish to override the domain name set by `hostconfig`, type the substitute domain name in the file `/etc/defaultdomain`.

6. **Ensure that the search paths in the client's `/etc/nsswitch.conf` reflects the name service requirements for your network.**

### ▼ How to Specify a Router for the Network Client

1. **If you have only one router on the network and you want the network configuration server to automatically specify its name, ensure that the network client does not have a `/etc/defaultrouter` file.**

2. **To override the name of the default router provided by the network configuration server:**

   a. **Create `/etc/defaultrouter` on the network client.**

   b. **Type the host name and IP address of the machine you have designated as the default router.**

   c. **Add the host name and IP address of the designated default router to the network client's `/etc/inet/hosts`.**

3. **If you have multiple routers on the network, create `/etc/defaultrouter` on the network client, but leave it empty.**

Creating `/etc/defaultrouter` and leaving it empty causes one of the two dynamic routing protocols to run: ICMP Router Discovery protocol (RDISC), or Routing Information Protocol (RIP). The system first runs the program `in.rdisc`, which looks for routers that are running the router discovery protocol. If it finds one such router, `in.rdisc` continues to run and keeps track of the routers that are running the RDISC protocol.

If the system discovers that routers are not responding to the RDISC protocol, it uses RIP and runs the daemon `in.routed` to keep track of them.

### *After Installing a Network Client*

After you have finished editing the files on each network client machine, do the following on the network configuration server.

1. **Add entries for the hosts in the `ethers` and `hosts` databases.**
2. **Add entries for the hosts to the `bootparams` database.**
   To simplify matters, you can type a wild card in the `bootparams` database in place of individual entries for each host. For an example, see ”bootparams Database” on page 67.
3. **Reboot the server.**

## *Configuring Standard TCP/IP Services*

Services such as `telnet`, `ftp`, and `rlogin` are started by the `inetd` daemon, which runs automatically at boot time. Like the name service ordering specified in `nsswitch.conf`, you can configure TCP/IP services in the file `/etc/inetd.conf` by using the `inetd -t` flag.

For example, you can use `inetd` to log the IP addresses of all incoming TCP connections (remote logins and `telnet`). To turn the logging on, kill the running `inetd` and type:

```
# /usr/sbin/inetd -t -s
```

The `-t` switch turns on TCP connection-tracing in `inetd`.

Refer to the `inetd` (1M) and `inetd.conf` (4) man pages.

See *NIS+ and FNS Administration Guide* and *NIS+ and DNS Setup and Configuration Guide* for further information on name services.

## *Overview of the Booting Processes*

The following information is provided for your reference. It is a brief overview of the network booting processes to help you better visualize what is happening during configuration.

**Note –** The names of startup scripts may change from one release to another.

1. You start the operating system on a host.
2. The kernel runs `/sbin/init`, as part of the booting process.
3. `/sbin/init` runs the `/etc/rcS.d/S30rootusr.sh`. startup script.
4. The script runs a number of system startup tasks, including establishing the minimum host and network configurations for diskless and dataless operations. `/etc/rcS.d/S30rootusr.sh` also mounts the `/usr` file system.
   a. If the local database files contain the required configuration information (host name and IP address), the script uses it.
   b. If the information is not available in local host configuration files, `/etc/rcS.d/S30rootusr.sh` uses RARP to acquire the host's IP address.
5. If the local files contain domain name, host name, and default router address, the machine uses them. If the configuration information is not in local files, then the system uses the Bootparams protocol to acquire the host name, domain name, and default router address. Note that the required information must be available on a network configuration server that is located on the same network as the host. This is necessary because no internetwork communications exist at this point.
6. After `/etc/rcS/S30rootusr.sh` completes its tasks and several other boot procedures have executed, `/etc/rc2.d/S69inet` runs. This script executes startup tasks that must be completed before the name services (NIS, NIS+, or DNS) can start. These tasks include configuring the IP routing and setting the domain name.
7. At completion of the `S69inet` tasks, `/etc/rc2.d/S71rpc` runs. This script starts the NIS, NIS+, or DNS name service.
8. After `/etc/rc2.d/S71` runs, `/etc/rc2.d/S72inetsvc` runs. This script starts up services that depend on the presence of the name services. `S72inetsvc` also starts the daemon `inetd`, which manages user services such as `telnet`.

See *System Administration Guide, Volume I and II* for a complete description of the booting process.

# 4

# Configuring Routers 5

This chapter describes routing protocols and contains procedures specifically for configuring routers on TCP/IP networks. A router is any machine that has two or more network interfaces and forwards packets from one network to another. Two common types of routers are computers with additional network interfaces in their card slots and dedicated routers sold by various manufacturers.

This chapter does not explain the theory of routing.You can find that information in ”Network Topology” on page 40 and ”How Routers Transfer Packets” on page 42 explain basic topics regarding routing. Tasks for creating subnets are found in ”netmasks Database” on page 56.


| | |
|---|---|
| *Routing Protocols* | *page 81* |
| *How to Configure a Machine as a Router* | *page 83* |
| *Creating a Multihomed Host* | *page 86* |


## Routing Protocols

Solaris system software supports two routing protocols: Routing Information Protocol (RIP) and ICMP Router Discovery (RDISC). RIP and RDISC are both standard TCP/IP protocols.

### *Routing Information Protocol (RIP)*

RIP is implemented by `in.routed`, the routing daemon, which automatically starts when the machine boots. When run on a router with the `-s` option specified, `in.routed` fills the kernel routing table with a route to every reachable network and advertises “reachability” through all network interfaces.

When run on a host with the `-q` option specified, `in.routed` extracts routing information but does not advertise reachability. On hosts, routing information can be extracted in two ways:

- Do *not* specify the `-S` flag (capital “S”: “Space-saving mode”) and `in.routed` will build a full routing table exactly as it does on a router.
- Specify the `-S` flag and `in.routed` will create a minimal kernel table, containing a single default route for each available router.

### *ICMP Router Discovery (RDISC) Protocol*

Hosts used RDISC to obtain routing information from routers. Thus, when hosts are running RDISC, routers must also run another protocol, such as RIP, in order to exchange router information among themselves.

RDISC is implemented by `in.rdisc`, which should run on both routers and hosts. Normally, when `in.rdisc` runs on a host, it enters a default route for each router that is also running `in.rdisc`. Note that a host that is running `in.rdisc` will not discover routers that are running only RIP. Note also that when routers are running `in.rdisc` (rather than `in.routed`), they can be configured to have a different preference, which causes hosts to select a better router. See the `rdisc`(1M) man page.

## *Configuring Routers*

TCP/IP’s first requirement for a router is that the machine must have at least two network interfaces installed, as introduced in ”Network Interfaces” on page 5. As long as one of the network interfaces is not disabled, the router will automatically “talk” to the RDISC and RIP protocols. These protocols keep track of routers on the network and advertise the router to the hosts on the network.

After the router is physically installed on the network, configure it to operate in local files mode, as described in "How to Configure a Host for Local Files Mode" on page 72. This assures that routers will boot in case the network configuration server is down. Remember that unlike a host, a router has at least two interfaces to configure.

## *Configuring Both Router Network Interfaces*

Since a router provides the interface between two or more networks, you must assign a unique name and IP address to each of the router's network interface cards. Thus, each router will have a host name and IP address associated with its primary network interface, plus at least one more unique name and IP address for each additional network interface.

### ▼ How to Configure a Machine as a Router

Become superuser on the machine that will be configured as a router and do the following:

1. **Create an `/etc/hostname.`*interface* file or each network interface installed.**
   For example, create `hostname.ie0` and `hostname.ie1`. (See "/etc/hostname.interface File" on page 51 for more information.)

2. **Type in each file the host name you have selected for that interface.**
   For example, you could type the name `timbuktu` in the file `hostname.ie0` and then type the name `timbuktu-201` in the file `hostname.ie1`. Both interfaces would be located on the same machine.

3. **Type the host name and IP address of each interface into `/etc/inet/hosts`.**
   For example:

```
192.9.200.20   timbuktu          #interface for network 192.9.200
192.9.201.20   timbuktu-201      #interface for network 192.9.201
192.9.200.9     gobi
192.9.200.10    mojave
192.9.200.110   saltlake
192.9.200.12    chilean
```

   The interfaces `timbuktu` and `timbuktu-201` are on the same machine. Note that the network address for `timbuktu-201` is different than that of `timbuktu`. That is because the medium for network 192.9.201 is connected to the `timbuktu-201` network interface while the media for network 192.9.200 is connected to the `timbuktu` interface.

4. **If router is connected to any subnetted network, edit `/etc/inet/netmasks` and type the local network number (129.9.0.0, for example) and associated netmask number (255.255.255.0, for example).**

## *How a Machine Determines If It Is a Router*

The `/etc/rc2.d/S69inet` startup script, which runs when the machine boots, determines whether a machine is a router or a host. This decision also determines whether the routing protocols (RIP and RDISC) should run in router mode or host mode.

The `/etc/rc2.d/S69inet` script concludes that a machine is a router if the following two conditions exist:

- More than one `/etc/hostname.`*interface* file exists.
- More than one interface was configured “up” by the `ifconfig` command. (See the `ifconfig`(1M) man page.)

If only one interface is found, the script concludes that the machine is a host. See ”Configuring Both Router Network Interfaces” on page 83. Any interface that is configured by any means other than an `/etc/hostname.`*interface* file will not be taken into account.

## Automatic Routing Protocol Selection

The startup script then must determine whether to start up a routing protocol (RIP or RDISC) on the machine or use static routing

### To Select Static Routing on a Host

If the host is a diskless client or network client, you simply add an entry for a router on the network into `/etc/defaultrouter`. (See "/etc/defaultrouter File" on page 52.) A single static default route is then installed in the routing table. Under this condition, the host will not run any dynamic routing protocol (such as RIP and RDISC).

### To Select Dynamic Routing on a Host

To force a diskless client or network client to select a dynamic routing protocol, its `/etc/defaultrouter` file should be empty. The type of dynamic routing used is selected according to the following criteria:

- If the `/usr/sbin/in.rdisc` program exists, the startup script will start `in.rdisc`. Any router on the network that is running RDISC will then respond to any RDISC queries from the host. If at least one router responds, the host will select RDISC as its routing protocol.
- If the network router is not running RDISC or fails to respond to the RDISC queries, then `in.rdisc` on the host exits. The host then starts `in.routed`, which runs RIP.

## Forcing a Machine to Be a Router

You can force a machine that has only one `/etc/hostname.`*interface* file (by default a host) to be a router. To do so, create a file named `/etc/gateways` and leave it empty. This is important if you decide to configure PPP links, as explained in "Routing Considerations" on page 126.

## *Creating a Multihomed Host*

By default, TCP/IP considers any machine with multiple network interfaces to be a router. However, you can change a router into a *multihomed host*—a machine with more than one network interface that does not run routing protocols or forward IP packets. You typically configure the following types of machines as multihomed hosts:

- **NFS servers**, particularly large data centers, can be attached to more than one network in order to share files among a large pool of users. These servers don't need to maintain routing tables.
- **Database servers** can have multiple network interfaces for the same reason as NFS servers—to provide resources to a large pool of users.
- **Firewall gateways** are machines that provide the connection between a company's network and public networks such as the Internet. Administrators set up firewalls as a security measure. When configured as a firewall, the host will not pass packets between the networks attached to it. On the other hand, it can still provide standard TCP/IP services, such as `ftp` or `rlogin`, to authorized users.

Since TCP/IP considers any machine with multiple network interfaces to be a router, you need to perform a few operations to turn it into a multihomed host.

### ▼ How to Create a Multihomed Host

Become superuser on the prospective multihomed host and do the following:

1. **Create an `/etc/hostname.`*interface* file for each additional network interface installed in the machine.**
2. **Type**:

```
% touch /etc/notrouter
```

   This creates an empty file called `/etc/notrouter`.

3. **Reboot the machine.**

When the machine reboots, the startup script looks for the presence of the `/etc/notrouter` file. If the file exists, the startup script does not run `in.routed -s` or `in.rdisc -r`, and does not turn on IP forwarding on all interfaces configured "up" by `ifconfig`. This will happen regardless of whether an `/etc/gateways` file exists. Thus the machine is now a multihomed host.

## *Turning On Space-Saving Mode*

Space-saving mode provides the host with a table that contains only the default routes. On a host, `in.routed` runs with space saving mode turned off by default.

If you do not want the host to have a full routing table (which provides increased protection against misconfigured routers), turn space saving mode on. To do so, edit the `/etc/rc2.d/S69inet` startup script by changing the line:

```
/usr/sbin/in.routed -q
```

to

```
/usr/sbin/in.routed -q -S
```

## *Turning Off ICMP Router Discovery on the Host*

For reasons involving router reliability, you may not want your hosts to use RDISC. To turn RDISC off, change the name of the host's `/usr/sbin/in.rdisc` to some other name, such as `/usr/sbin/in.rdisc.saved`, and then reboot the host.

## *Turning Off ICMP Router Discovery on the Router*

If the automatic selection of RIP rather than RDISC by a host is to work reliably, the routers in the network (particularly those running RDISC) also must work reliably.

If your routers are not running RDISC and you install a single Solaris router, by default all hosts connected to that router will rely on it alone. To have the hosts on that network use the other routers as well, turn off RDISC on the new router. To do this, change the name of the router's `/usr/bin/in.rdisc` file to some other file name and reboot the router.

# *Troubleshooting TCP/IP* 6

This chapter describes general methods for troubleshooting TCP/IP networks and some of the tools available for doing so. These tools include `ping`, `ifconfig`, `netstat`, and `route`.


| | |
|---|---|
| *General Troubleshooting Methods* | *page 89* |
| *Running Software Checks* | *page 90* |
| *ping Command* | *page 90* |
| *ifconfig Command* | *page 92* |
| *netstat Command* | *page 93* |
| *Logging Network Problems* | *page 97* |
| *Displaying Packet Contents* | *page 97* |


## *General Troubleshooting Methods*

One of the first signs of trouble on the network is a loss of communications by one or more hosts. If a host refuses to come up at all the first time it is added to the network, the problem may lie in one of the configuration files, or in the network interface. If a single host suddenly develops a problem, the network interface may be the cause. If the hosts on a network can communicate with each other but not with other networks, the problem could lie with the router, or it could lie in another network.

# 6

You can use the `ifconfig` program to obtain information on network interfaces and `netstat` to display routing tables and protocol statistics. Third-party network diagnostic programs provide a number of troubleshooting utilities. Refer to third-party documentation for information.

Less obvious are the causes of problems that degrade performance on the network. For example, you can use tools like `ping` to quantify problems like the loss of packets by a host.

## *Running Software Checks*

If there is trouble on the network, some actions that you can take to diagnose and fix software-related problems include

1. Using the `netstat` command to display network information.
2. Checking the `hosts` database to make sure that the entries are correct and up to date.
3. If you are running RARP, checking the Ethernet addresses in the `ethers` database to make sure that the entries are correct and up to date.
4. Trying to connect by `telnet` to the local host.
5. Ensuring that the network daemon `inetd` is running. To do this, log in as superuser and type:

```
# ps -ef | grep inetd
```

Here is an example of output displayed if the `inetd` daemon is running:

```
root 57 1 0 Apr 04 ? 3:19 /usr/sbin/inetd -s
root 4218 4198 0 17:57:23 pts/3 0:00 grep inetd
```

## `ping` *Command*

Use the `ping` command to find out whether there is IP connectivity to a particular host. The basic syntax is:

`/usr/sbin/ping` *host [timeout]*

where *host* is the host name of the machine in question. The optional *timeout* argument indicates the time in seconds for `ping` to keep trying to reach the machine—20 seconds by default. The `ping`(1M) man page describes additional syntaxes and options.

When you run `ping`, the ICMP protocol sends a datagram to the host you specify, asking for a response. (ICMP is the protocol responsible for error handling on a TCP/IP network. See ”ICMP Protocol” on page 15 for details.)

Suppose you type:

```
$ ping elvis
```

If host `elvis` is up, this message is displayed:

```
elvis is alive
```

indicating that `elvis` responded to the ICMP request. However, if `elvis` is down or cannot receive the ICMP packets, you receive the following response from `ping`:

```
no answer from elvis
```

If you suspect that a machine may be losing packets even though it is up, you can use the `-s` option of `ping` to try and detect the problem. For example, type:

```
$ ping -s elvis
```

`ping` continually sends packets to `elvis` until you send an interrupt character or a timeout occurs. The responses on your screen will resemble:

```
PING elvis: 56 data bytes
64 bytes from 129.144.50.21: icmp_seq=0. time=80. ms
64 bytes from 129.144.50.21: icmp_seq=1. time=0. ms
64 bytes from 129.144.50.21: icmp_seq=2. time=0. ms
64 bytes from 129.144.50.21: icmp_seq=3. time=0. ms
.
.
.
----elvis PING Statistics----
4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms) min/avg/max = 0/20/80
```

The packet-loss statistic indicates whether the host has dropped packets.

If `ping` fails, check the status of the network reported by `ifconfig` and `netstat`, as described in ”ifconfig Command” and ”netstat Command” on page 93.

## `ifconfig` *Command*

The `ifconfig` command displays information about the configuration of an interface that you specify. (Refer to the `ifconfig`(1M) man page for complete details.) The syntax of `ifconfig` is:

`ifconfig` *interface-name [protocol_family]*

If you want information about a specific interface, for example `le0`, type:

```
$ ifconfig le0
```

For an `le0` interface, your output will resemble the following:

```
le0: flags=863<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
   inet 129.144.44.140 netmask ffffff00 broadcast 129.144.44.255
ether 8:0:20:8:el:fd
```

The flags section above shows that the interface is configured “up,” capable of broadcasting, and not using “trailer” link level encapsulation. The `mtu` field tells you that this interface has a maximum transfer rate of 1500. Information on the second line includes the IP address of the host you are using, the netmask being currently used, and the IP broadcast address of the interface. The third line gives the machine address (Ethernet, in this case) of the host.

A useful `ifconfig` option is `-a`, which provides information on all interfaces on your network. For example, typing

```
ifconfig -a
```

produces:

```
le0:  flags=49<UP,LOOPBACK,RUNNING> mtu 8232
     inet 127.144.44.140 netmask ff000000
le0:flags=863<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 129.144.44.140 netmask ffffff00 broadcast 129.144.44.255
ether 8:0:20:8:el:fd
```

Output that indicates an interface is not running might mean a problem with that interface. In this case, see the `ifconfig`(1M) man page.

## `netstat` *Command*

The `netstat` command generates displays that show network status and protocol statistics. You can display the status of TCP and UDP endpoints in table format, routing table information, and interface information.

`netstat` displays various types of network data depending on the command line option selected. These displays are the most useful for system administration. The syntax for this form is:

`netstat [-m] [-n] [-s] [-i | -r] [-f` *address_family*`]`

The most frequently used options for determining network status are: `-s`, `-r`, and `-i`. See the `netstat`(1M) man page for a description of the options.

## Displaying Per Protocol Statistics

The `netstat -s` option displays per protocol statistics for the UDP, TCP, ICMP, and IP protocols. The result resembles the display shown in the example below. (Parts of the output have been truncated.) The information can indicate areas where a protocol is having problems. For example, statistical information from ICMP can indicate where this protocol has found errors.

```
UDP
        udpInDatagrams      =  3928   udpInErrors         =     0
        udpOutDatagrams     =  2455
TCP
        tcpRtoAlgorithm     =     4   tcpRtoMin           =   200
        tcpRtoMax           = 60000   tcpMaxConn          =    -1
        tcpActiveOpens      =     4   tcpPassiveOpens     =     2
        tcpAttemptFails     =     3   tcpEstabResets      =     1
        tcpCurrEstab        =     1   tcpOutSegs          =   315
        tcpOutDataSegs      =   288   tcpOutDataBytes     = 10547
        tcpRetransSegs      =    29   tcpRetransBytes     =  8376
        tcpOutAck           =    27   tcpOutAckDelayed    =    23
        tcpOutUrg           =     2   tcpOutWinUpdate     =     2
        tcpOutWinProbe      =     0   tcpOutControl       =     8
        tcpOutRsts          =     0   tcpOutFastRetrans   =     1
        tcpInSegs           =   563
        tcpInAckSegs        =   289   tcpInAckBytes       = 10549
        tcpInDupAck         =    27   tcpInAckUnsent      =     0
        tcpInInorderSegs    =   254   tcpInInorderBytes   =   673
        tcpInUnorderSegs    =     0   tcpInUnorderBytes   =     0
        tcpInDupSegs        =     0   tcpInDupBytes       =     0
        tcpInPartDupSegs    =     0   tcpInPartDupBytes   =     0
        tcpInPastWinSegs    =     0   tcpInPastWinBytes   =     0
        tcpInWinProbe       =     0   tcpInWinUpdate      =   237
        tcpInClosed         =     0   tcpRttNoUpdate      =    21
        tcpRttUpdate        =   266   tcpTimRetrans       =    26
        tcpTimRetransDrop   =     0   tcpTimKeepalive     =     0
        tcpTimKeepaliveProbe=     0   tcpTimKeepaliveDrop =     0
IP
        ipForwarding        =     2   ipDefaultTTL        =   255
        ipInReceives        =  4518   ipInHdrErrors       =     0
        ipInAddrErrors      =     0   ipInCksumErrs       =     0
        ipForwDatagrams     =     0   ipForwProhibits     =     0
        ipInUnknownProtos   =     0   ipInDiscards        =     0
        ipInDelivers        =  4486   ipOutRequests       =  2805
```

```
        ipOutDiscards       =     5   ipOutNoRoutes       =     0
        ipReasmTimeout      =    60   ipReasmReqds        =     2
        ipReasmOKs          =     2   ipReasmFails        =     0
        ipReasmDuplicates   =     0   ipReasmPartDups     =     0
        ipFragOKs           =    20   ipFragFails          =     0
        ipFragCreates       =   116   ipRoutingDiscards   =     0
        tcpInErrs           =     0   udpNoPorts          =    33
        udpInCksumErrs      =     0   udpInOverflows      =     6
        rawipInOverflows    =     0
ICMP
        icmpInMsgs          =     0   icmpInErrors        =     0
        icmpInCksumErrs     =     0   icmpInUnknowns      =     0
        icmpInDestUnreachs  =     0   icmpInTimeExcds     =     0
        icmpInParmProbs     =     0   icmpInSrcQuenchs    =     0
        icmpInRedirects     =     0   icmpInBadRedirects  =     0
        icmpInEchos         =     0   icmpInEchoReps      =     0
        icmpInTimestamps    =     0   icmpInTimestampReps =     0
        icmpInAddrMasks     =     0   icmpInAddrMaskReps  =     0
        icmpInFragNeeded    =     0   icmpOutMsgs         =     7
        icmpOutDrops        =     5   icmpOutErrors       =     0
        icmpOutDestUnreachs =     1   icmpOutTimeExcds    =     0
        icmpOutParmProbs    =     0   icmpOutSrcQuenchs   =     6
        icmpOutRedirects    =     0   icmpOutEchos        =     0
        icmpOutEchoReps     =     0   icmpOutTimestamps   =     0
        icmpOutTimestampReps=     0   icmpOutAddrMasks    =     0
        icmpOutAddrMaskReps =     0   icmpOutFragNeeded   =     0
        icmpInOverflows     =     0
IGMP:
           0 messages received
           0 messages received with too few bytes
           0 messages received with bad checksum
           0 membership queries received
           0 membership queries received with invalid field(s)
           0 membership reports received
           0 membership reports received with invalid field(s)
           0 membership reports received for groups to which we
           belong
           0 membership reports sent
```

## *Displaying Network Interface Status*

The `-i` option of `netstat` shows the state of the network interfaces that are

configured with the machine where you ran the command. Here is a sample display produced by `netstat -i`.

```
Name   Mtu  Net/Dest      Address   Ipkts    Ierrs  Opkts    Oerrs  Collis Queue
le0    1500 b5-spd-2f-cm  tatra     14093893 8492   10174659 1119   2314178 0
lo0    8232 loopback      localhost 9299762  5442   12451748 0      775125  0
```

Using this display, you can find out how many packets a machine thinks it has transmitted and received on each network. For example, the input packet count (`Ipkts`) displayed for a server may increase each time a client tries to boot, while the output packet count (`Opkts`) remains steady. This suggests that the server is seeing the boot request packets from the client, but does not realize it is supposed to respond to them. This might be caused by an incorrect address in the `hosts` or `ethers` database.

On the other hand, if the input packet count is steady over time, it means that the machine does not see the packets at all. This suggests a different type of failure, possibly a hardware problem.

## *Displaying Routing Table Status*

The `-r` option of `netstat` displays the IP routing table. Here is a sample display produced by `netstat -r` run on machine `tenere`.

```
Routing tables
Destination   Gateway  Flags    Refcnt  Use    Interface
temp8milptp   elvis    UGH      0       0
irmcpeb1-ptp0 elvis    UGH      0       0
route93-ptp0  speed    UGH      0       0
mtvb9-ptp0    speed    UGH      0       0
              .
mtnside       speed    UG       1       567
ray-net       speed    UG       0       0
mtnside-eng   speed    UG       0       36
mtnside-eng   speed    UG       0       558
mtnside-eng   tenere   U        33      190248    le0
```

The first column shows the destination network, the second the router through which packets are forwarded. The `U` flag indicates that the route is up; the `G` flag indicates that the route is to a gateway. The `H` flag indicates that the destination is a fully qualified host address, rather than a network.

The `Refcnt` column shows the number of active uses per route, and the `Use` column shows the number of packets sent per route. Finally, the `Interface` column shows the network interface that the route uses.

## *Logging Network Problems*

If you suspect a routing daemon malfunction, you can log its actions, including all packet transfers. To create a log file of routing daemon actions, supply a file name when you start up the `routed` daemon. For example:

```
# /usr/sbin/in.routed /var/routerlog
```

**Caution –** On a busy network, this may generate almost continuous output.

## *Displaying Packet Contents*

`snoop` captures network packets and displays their contents. Packets can be displayed as they are received, or saved to a file. For complete information, refer to the `snoop`(1M) man page.

# 6

# *Part 2 — Expanding Your Network With PPP*

Part 2 explains how to set up and administer asynchronous PPP communications links on the network. The text assumes that you are an experienced network administrator who is familiar with TCP/IP.


| | |
|---|---|
| *PPP Network Interfaces* | *page 103* |
| *Point-to-Point Communications Links* | *page 103* |
| *Multipoint Communications Links* | *page 109* |
| *Determining IP Addressing for Your PPP Link* | *page 123* |
| *Routing Considerations* | *page 126* |
| *Checklist for Configuring PPP* | *page 128* |
| *Installing the PPP Software* | *page 132* |
| *Editing the /etc/asppp.cf Configuration File* | *page 139* |
| *How to Manually Start PPP* | *page 146* |
| *Checking Interface Status* | *page 150* |
| *Checking Connectivity* | *page 151* |
| *Configuring Dynamically Allocated PPP Links* | *page 165* |
| *Configuring a Virtual Network* | *page 172* |
| *Configuration Keywords* | *page 180* |

# Understanding PPP 7

This chapter presents an overview of Solaris PPP, a data-link protocol included in the TCP/IP protocol suite. The text includes product specifications, introductions to the most typical PPP configurations, and definitions of the terms related to PPP.


| | |
|---|---|
| *Solaris PPP Specifications* | *page 102* |
| *PPP Network Interfaces* | *page 103* |
| *Point-to-Point Communications Links* | *page 103* |
| *Dial-in Server With Dynamic Point-to-Point Link* | *page 107* |
| *Multipoint Communications Links* | *page 109* |
| *Introducing the PPP Software* | *page 111* |


## Overview of Solaris PPP

PPP enables you to connect computers and networks at separate physical locations by using modems and telephone lines. With PPP, users with computers at home or in remote offices can connect to your site's network. You can also use the combination of PPP software, a modem, and telephone lines as a router connecting networks in different places. PPP offers strategies for configuring these machines and networks, which are introduced in this chapter.

# 7

## Solaris PPP Specifications

Solaris PPP is an asynchronous implementation of the standard data-link level Point-to-Point Protocol (PPP) included in the TCP/IP protocol suite and provided by a number of router system vendors and terminal concentrators. It includes a standard encapsulation protocol, making datagram transmission transparent to network layer protocols.

The major characteristics of the Solaris PPP protocol are:

- Implements the Internet Point-to-Point Protocol, as defined in RFC 1331
- Provides error detection through CRC
- Supports full duplex transmission

The major functions of the protocol are:

- Interface for IP to forward packets over asynchronous serial lines
- Connection establishment on demand
- Configurable options negotiation
- Connection termination (automatic hang-up)

## Transmission Facilities Used by PPP

PPP supports interfaces to RS-232-C (V.24) facilities through the CPU serial ports included on most machines running the Solaris software. In addition, PPP will run over optional asynchronous serial ports supplied or supported by many manufacturers of machines that run the Solaris software. PPP supports the maximum data rates that your machine's serial ports can achieve. Consult the manufacturer of your computer system for more details on the speeds supported by your machine's serial hardware.

**Note –** Machines of the x86 architecture require UARTs that run above a certain speed. See the *x86 Device Configuration Guide* for details.

## Standards Conformance

PPP, and the routing functions in the Solaris software, use industry-standard conventions for performing their tasks. These conventions support:

- Forwarding of IP datagrams

- Reception of packets for forwarding from any IP-compatible networked system
- Delivery of packets to any IP-compatible networked system on local-area network media, such as Ethernet, Token Ring, and FDDI
- Use of standard routing protocols, enabling users to exchange packets with equipment that supports the PPP protocol from many manufacturers

## *PPP Network Interfaces*

PPP enables asynchronous devices, such as modems, to become network interfaces. Solaris PPP enables you to configure two types of *virtual network interfaces*, `ipdptp`*n* and `ipd`*n*. (The letter *n* represents the device number you assign to the interface.)

PPP network interfaces are considered virtual network interfaces because they do not involve network hardware, as does, for example, an Ethernet interface. Moreover, they are not associated with any particular serial port. The PPP network interfaces reside in the `/devices` directories along with the physical network interfaces. (For information on physical network interfaces, see ”Network Interfaces” on page 5.)

The type of network interface you use depends on the PPP communications link you want to set up. The `ipdptp` interface supports point-to-point PPP links; the `ipd` interface supports point-to-multipoint links (called “multipoint links”).

## *Extending Your Network With PPP*

This section introduces PPP-related communications concepts used. It also explains the most typical PPP configurations that you are likely to set up.

### *Point-to-Point Communications Links*

The most common use of Solaris PPP is to set up a point-to-point communications link. A generic point-to-point communications configuration consists of two endpoints connected by a communications link. In a generic configuration, an *endpoint* system could be a computer or terminal, either in an isolated location or physically connected to a network. The term *communications link* refers to the hardware and software connecting these endpoint systems. Figure 7-1 on page 104 illustrates these concepts.

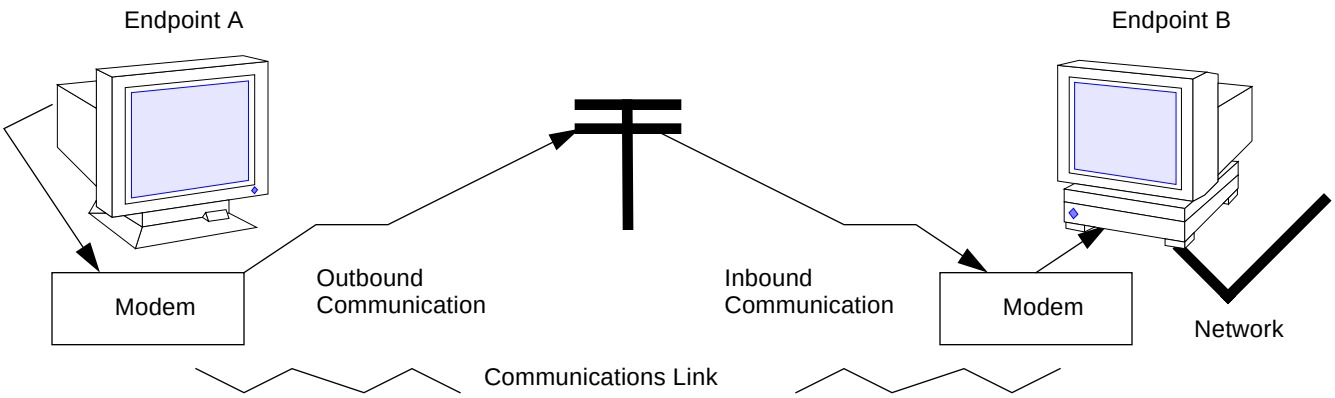



*Figure 7-1* Basic Point-to-Point Link

## *Dial-out Operations and Outbound Communications*

When an endpoint system wants to communicate with the end point on the other side of the communications link, it begins a *dial-out operation*. For example, to communicate with endpoint B, a user at its peer host, endpoint A, types `rlogin` *end-point-B*. This causes endpoint A to dial out over the communications link. In this instance, endpoint A functions as a *dial-out* machine. The `rlogin` command causes its modem to dial the phone number of endpoint B. The action it starts and information it passes are considered *outbound communication*s.

## *Dial-ins and Inbound Communications*

When the data travels over the link to endpoint B, this system receives incoming data and sends an acknowledgment signal to endpoint A to establish communications. In this instance, endpoint B functions as a *dial-in* machine, since it permits other systems to dial in to it. The information passed to the communications recipient and the actions the recipient takes are considered *inbound communication*s.

## Point-to-Point Configurations Supported by Solaris PPP

Solaris PPP supports four types of point-to-point configurations:

- Host in one location connected to a host at another physical location, as shown in Figure 7-1 on page 104
- Dial-in servers with dynamic point-to-point links to remote hosts, as shown in Figure 7-2 on page 106
- Network connected to another physically distant network, as shown in Figure 7-3 on page 108
- Computers connected to a multipoint dial-in server physically attached to a distant network, as shown in Figure 7-4 on page 109

These PPP links provide essentially the same type of connectivity provided by a local area network but without broadcast capability. The sections below summarize the configuration types; Chapter 8, “Preparing Your PPP Configuration,” gives information for setting up each configuration type.

### Two Isolated Hosts Connected by a Point-to-Point Link

PPP enables you to set up a point-to-point link to connect two standalone machines in separate locations, effectively creating a network consisting solely of these two machines. This is the simplest point-to-point configuration because it involves only the two endpoints. The generic configuration shown in Figure 7-1 on page 104 also uses the host-to-host configuration.

### Nomadic Machines Connected to a Dial-in Server

In the past, standard dial-up or temporary connections permitted only ASCII terminals to connect to a network. With Solaris PPP, an individual machine can become part of a physically distant network by configuring it as one endpoint of the PPP link. The advantage of this *nomadic* connection is particularly apparent if your network includes users who travel frequently or work from home.

Figure 7-2 on page 106 shows nomadic computers, each with a point-to-point link to an endpoint system on the network. The endpoint on the network is a dial-in server.

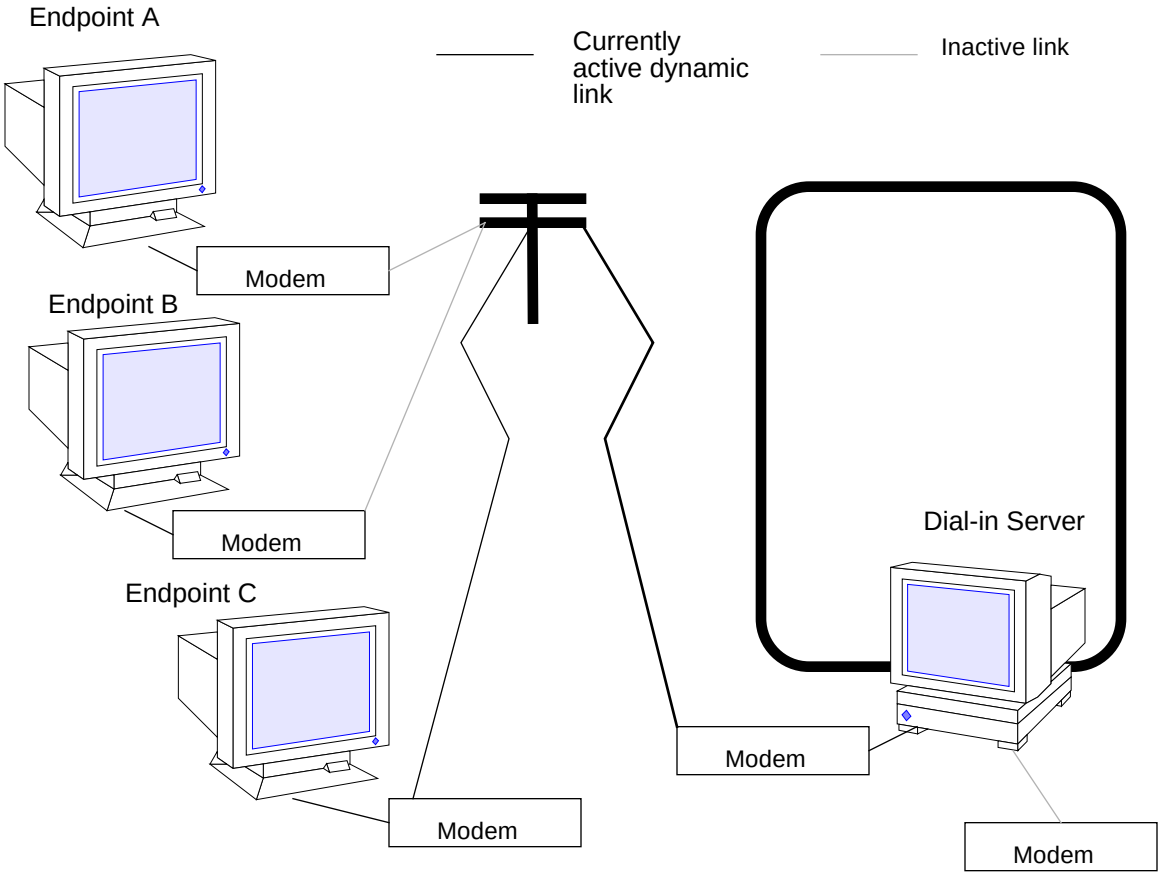



*Figure 7-2* Nomadic Computers and Dynamic Link Dial-in Server

### *Dial-in Server With Dynamic Point-to-Point Link*

The endpoint machine on the network shown in Figure 7-2 on page 106 functions as a dial-in server with dynamic point-to-point links. It is called a *dial-in server* because remote machines can dial in to it to reach the network. When the server receives a request to dial in from a machine, the server allocates the PPP link to the machine on an as-needed basis.

A dial-in server can communicate with the remote hosts through a dynamic point-to-point link or through a multipoint link, as explained in ”Multipoint Communications Links” on page 109. The dynamic point-to-point link has the advantages of point-to-point communications: RIP can run over the link, and broadcasting is enabled. Perhaps most importantly, more than one machine on the physical network can function as the dial-in server. This allows you to configure backup servers, thus enabling redundancy and easier administration. Although the machines in Figure 7-2 can directly communicate with the network endpoint, they cannot directly communicate with each other. They must pass information to each other through the dial-in server endpoint.

### *Two Networks Connected by Point-to-Point Link*

You can use PPP to connect two separate networks through a point-to-point link, with one system on each network serving as an endpoint. These endpoints communicate through modems and phone lines, essentially in the same fashion as shown in Figure 7-1 on page 104. But in this setup, the endpoints, modems, and PPP software become routers for their physical networks. Using this type of configuration scheme, you can create an internetwork with wide geographic reach.

Figure 7-3 on page 108 shows two networks in different locations connected by a point-to-point link.

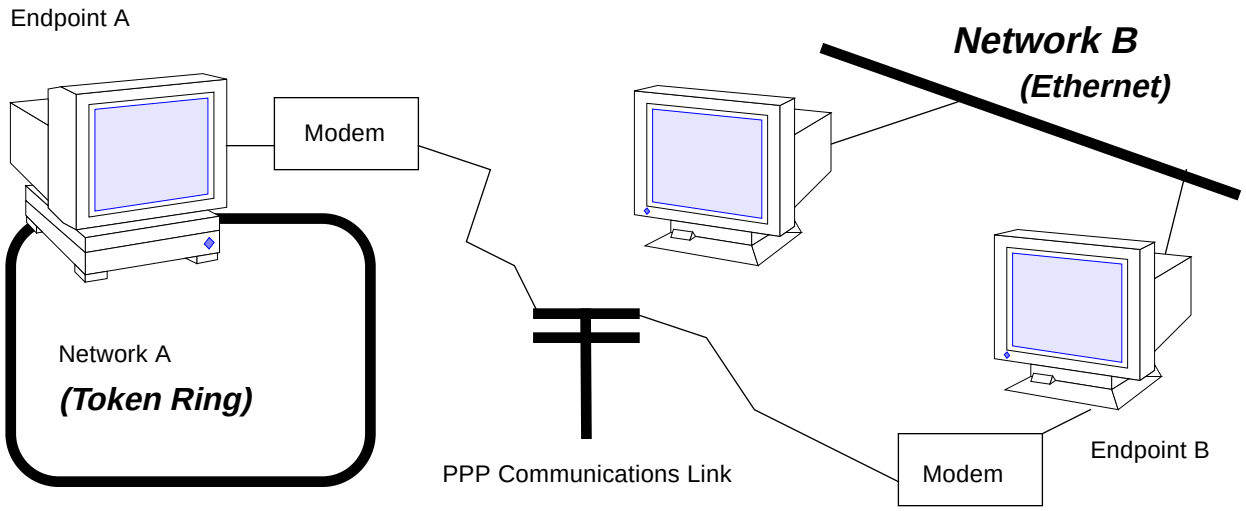



*Figure 7-3* Two Networks Connected by a PPP Link

In this example, endpoints A and B, their modems, public telephone lines, and the PPP software act as a router between the networks. These networks may have other hosts serving as routers between physical networks. Sometimes, the host functioning as the PPP router may have an additional network interface board, thus also serving as a router for a physical network.

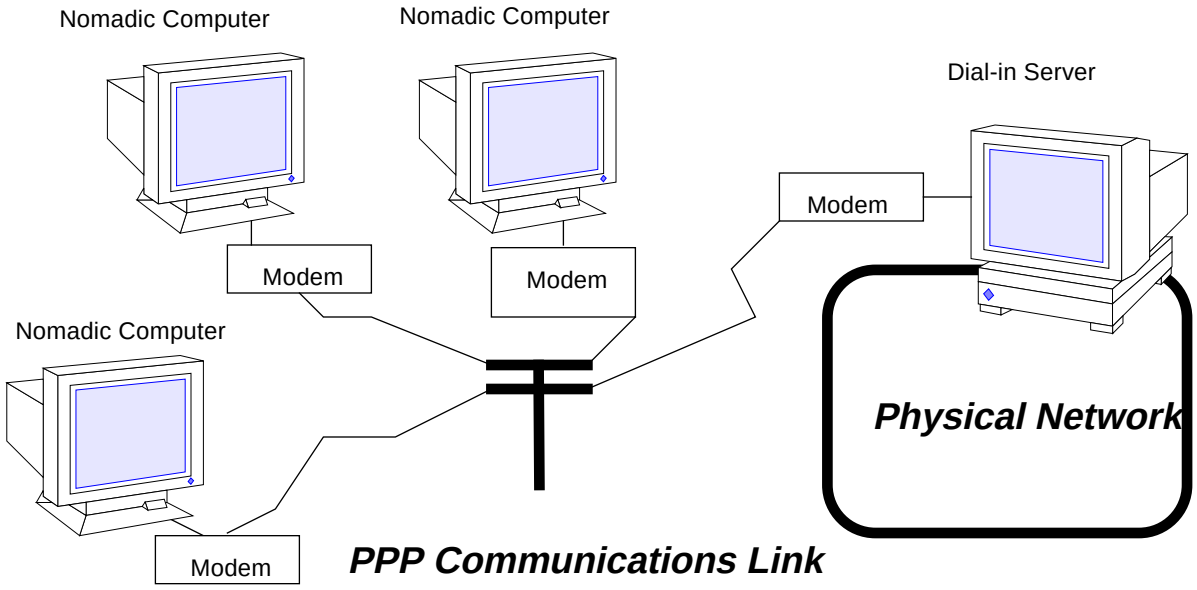



*Figure 7-4* Nomadic Computers and Multipoint Dial-in Server

## *Multipoint Communications Links*

You can use Solaris PPP to set up a multipoint communications link. In this type of configuration, an individual machine functions as one endpoint on the communications link. At the other end of the link may be several endpoint machines. This differs from point-to-point configurations, with a single endpoint system at either side of the communications link.

## *Multipoint Configurations Supported by PPP*

Two types of multipoint links you can configure with PPP are

- Dial-in server with multipoint connections to remote machines, as shown in Figure 7-4

- Logical, or *virtual*, network consisting of three or more nomadic computers as shown in Figure 7-5 on page 111

The sections below summarize these configurations; Chapter 8, “Preparing Your PPP Configuration,” explains how to set up the configuration.

## *Multipoint Dial-in Servers*

Figure 7-4 on page 109 shows three geographically isolated computers communicating through a point-to-point link to an endpoint machine on a network. However, the network endpoint machine can communicate with the nomadic computers through a *multipoint* link, thus making it a multipoint dial-in server. (You can also set up a dial-in server with dynamic point-to-point connections, as explained on page 107.)

The dial-in server can communicate with all the machines on the other end of its multipoint PPP link. Though the machines in Figure 7-4 can directly communicate with the multipoint dial-in server, they cannot communicate directly with each other. They must pass information to each other through the dial-in server.

## *Virtual Networks*

You can use PPP to set up a *virtual network* wherein the modems, PPP software, and telephone wires become the “virtual” network media. In a physical network, such as Ethernet or Token Ring, computers are directly cabled to the network media. In a virtual network, no true network media exist.

Machines become peer hosts on the virtual network when you configure each with a multipoint communications link. Then each host can dial out through its modem over phone lines to reach another endpoint machine. Each computer also functions as a dial-in machine, permitting its peer hosts on the virtual network to dial in to it.

Figure 7-5 on page 111 depicts a virtual network consisting of nomadic computers connected to each other through modems and telephone lines.

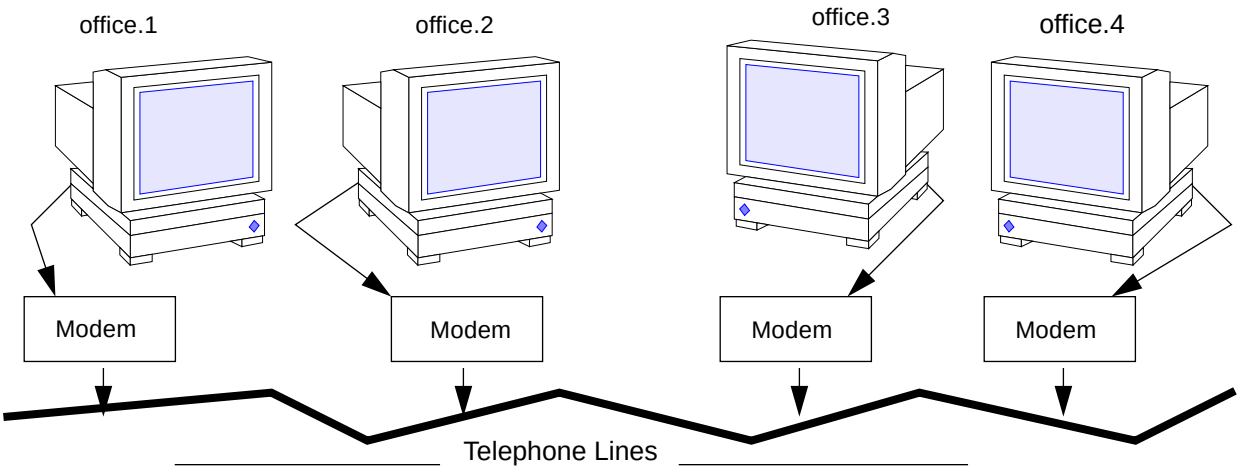



*Figure 7-5* Virtual Network of Nomadic Computers

Each machine exists in a different office, perhaps in a different town from other members of the virtual network. However, each machine can establish communications with its peer hosts over its multipoint communications links.

## *Introducing the PPP Software*

The PPP component software includes the

- Link manager (`/usr/sbin/aspppd`)
- Log in service (`/usr/sbin/asppp1s`)
- Configuration file (`/etc/asppp.cf`)
- Log file (`/var/adm/log/asppp.log`)
- FIFO file (`/tmp/.asppp.fifo`)

After you install the PPP software, you will find the `/etc/init.d/asppp` file, which is the run-control script for PPP. It is linked to several other files in the run-control directories.

Figure 7-6 on page 112 shows the software components of PPP and how they interact.

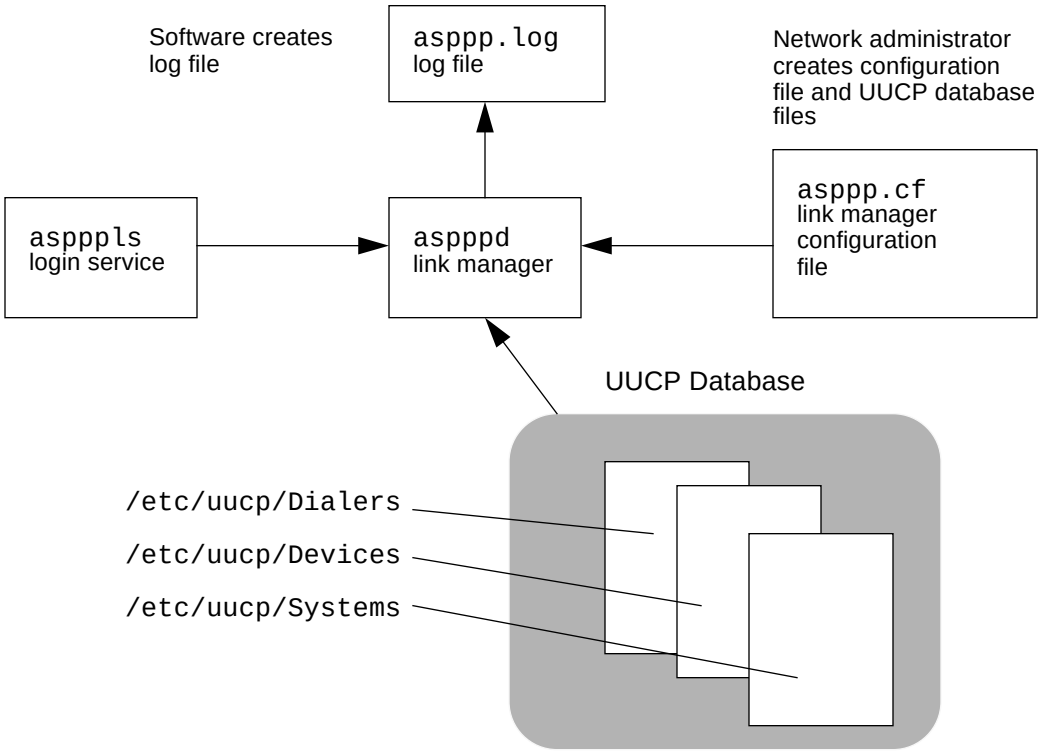



*Figure 7-6* PPP Component Software

## *Link Manager*

The `/usr/sbin/aspppd` link manager is a user-level daemon that automates the process of connecting to a remote host when PPP service is required. This automated process starts whenever any activity that generates IP traffic takes place (for example, a user logs in to a remote machine, accesses an NFS-mounted file, and so on). If a remote host tries to establish a connection, the link manager on the local host will complete the connection.

Refer to the `aspppd` (1m) man page for specific information about the link manager.

## Login Service

The /usr/sbin/aspppls login service is invoked as a login shell that starts PPP after you dial up and log in. Its function is similar to the /usr/lib/uucp/uucico command described in "Software Comprising UUCP" on page 186. When configuring a machine as a dial-in server, you must specify aspppls as the login shell in the /etc/passwd file in the entries for every nomadic computer allowed to dial in to the local host.

## Configuration File

The asppp.cf file provides the link manager with information about each remote endpoint with which the local host will communicate. You define this information in a section of the configuration file called a *path*. The path section also defines the PPP interface to be used and, optionally, other attributes determining how communications will take place, including security issues. "Parts of Basic Configuration File" on page 140 explains the sections of the asppp.cf file in detail.

*Code Example 7-1* An Unmodified asppp.cf File

```
#ident"@(#)asppp.cf1093/07/07 SMI"
#
# Copyright (c) 1993 by Sun Microsystems, Inc.
#
# Sample asynchronous PPP /etc/asppp.cf file
#
#

ifconfig ipdptp0 plumb mojave gobi private up

path
    inactivity_timeout 120    # Approx. 2 minutes
    interface ipdptp0
    peer_system_name Pgobi    # The name this system logs in with when
                              # it dials this server
                              # *OR* the entry we look up in
                              # /etc/uucp/Systems when we dial out.
```

### Log File

The link manager produces messages and logs them in the log file `/var/adm/log/asppp.log`. The level of detail reported into the file is controlled by the `-d` option of `aspppd` or the `debug_level` keyword in the configuration file. See ”Configuration Keywords” on page 180 and the `aspppd` (1m) man page for more information.

### FIFO File

The PPP FIFO file `/tmp/.asppp.fifo` is a named pipe used to communicate between `aspppd` and `asppp1s`. This file must be present in `/tmp` for the PPP login service to connect to the link manager. The `/tmp/.asppp.fifo` file is created, managed, and deleted by the link manager.

### UUCP Databases

Besides its component software, Solaris PPP uses information in three UUCP files, `/etc/uucp/Systems`, `/etc/uucp/Dialers`, and `/etc/uucp/Devices`, to help it establish the communications link. You must modify these files to enable a host to dial out over the PPP link. Alternatively, you can use the file `/etc/uucp/Sysfiles` to specify different names for the `Systems`, `Devices`, and `Dialers` files.

Refer to Chapter 12, “UUCP Databases and Programs,” for full descriptions of these UUCP files.

## How the Components Work Together

This section describes how the components of PPP function for outbound and inbound connections.

### Outbound Connections Scenario

Outbound communications begin when a user on one endpoint host initiates an activity involving the peer host on the other end of the PPP link. The following activities take place when a user types an `rcp` command to copy a file from a host on the other side of the link

1. `rcp` sends the data through the levels of the TCP/IP protocol stack.
2. A virtual network interface (`ipd`*n* or `ipdptp`*n)* receives the data in the form of IP packets.
3. The interface sends the `aspppd` link manager a connection request that initiates an outbound connection.
4. The link manager then:
   a. Verifies that the connection request corresponds to a configured path in the `/etc/asppp.cf` configuration file.
   b. Consults the UUCP database files (`/etc/uucp/Systems`, `/etc/uucp/Devices`, and `/etc/uucp/Dialers`) for specific information about the modem and destination system.
   c. Places a phone call to the destination host or attaches to the appropriate hardwired serial line.
5. The physical link to the peer host is established.
6. The link manager configures and initiates PPP.
7. The data-link layer is established, and the PPP modules on the peer hosts start communicating.
8. The link manager enables IP over the link.

The link manager then monitors the connection until an event such as an idle time out, line disconnect, or error condition occurs. When any of these events occur, the link manager disconnects from the peer host and returns to the idle state.

## *Inbound Connections Scenario*

The host initiating the inbound communication logs in, which invokes the `/usr/sbin/aspppls` login service. Then the following events occur:

1. The login service connects to the link manager through the `/tmp/.asppp.fifo` file.
2. The login service provides the link manager with information such as the login name used by the endpoint at the other end of the link.

3. The link manager uses this login name to find a corresponding configured path in the configuration file.
4. The link manager then configures and initiates PPP.
5. The data-link layer is established, and the PPP modules on the peer hosts start communicating.
6. The link manager enables IP over the link.

The link manager then monitors the connection until an event occurs such as an idle time out, line disconnect, or error condition. When any of these events occur, the link manager disconnects from the peer and returns to the idle state.

## *PPP Security*

Once you have completed installing PPP on every machine involved in your configuration, you can add either one or two levels of security for the PPP link.

The first level, Password Authentication Protocol (PAP), is the least secure. A password is sent over the circuit “in the clear” until authentication is acknowledged or the connection terminated.

The second level of security, Challenge-Handshake Authentication Protocol (CHAP), periodically verifies the identity of the peer—the other end of the point-to-point link. A challenge message is sent to the peer by the authenticator—the system starting the link or challenge. The response is checked against a “secret” not sent over the link, and if the values match, authentication is acknowledged. Otherwise, the link is terminated. The process of adding PPP security is described in “Editing asppp.cf for PAP/CHAP Security” on page 176.

# *Preparing Your PPP Configuration* 8

Before configuring the PPP software, you need to prepare the hardware and software involved and gather some information that is needed during the configuration process. This chapter explains the tasks you have to perform prior to configuration, such as

- Determining your network addressing scheme
- Ensuring that your hardware meets the requirements for PPP
- Preparing your software to meet the requirements for PPP

The chapter concludes with a checklist to help you organize this information before you configure your PPP link.


| | |
|---|---|
| *Remote Computer-to-Network Configuration* | *page 118* |
| *Remote Host-to-Remote Host Configuration* | *page 119* |
| *Network-to-Network Configuration* | *page 120* |
| *Dial-in Server With Dynamic Point-to-Point Links* | *page 121* |
| *Multipoint Dial-in Server* | *page 122* |
| *Hosts on a Virtual Network* | *page 123* |
| *Determining IP Addressing for Your PPP Link* | *page 123* |
| *Assigning a Network Number to the PPP Link* | *page 126* |
| *Turning Off RIP* | *page 127* |
| *PPP Hardware Requirements* | *page 127* |

# 8

## Determining Requirements for Your Configuration Type

Solaris PPP supports many configuration options, including:

- Remote computer-to-network over a point-to-point link
- Remote computer-to-remote computer over a point-to-point link
- Network-to-network over a point-to-point link
- Dial-in server-to-multiple remote computers through one or more dynamic point-to-point links
- Dial-in server-to-multiple remote computers through a multipoint link
- Multiple remote computers comprising a virtual network, all communicating through multipoint links

These configurations are introduced in “Extending Your Network With PPP” in Chapter 7, “Understanding PPP.”

This section describes the information you need to gather and tasks you have to perform for each configuration type before beginning the configuration process. Read the section that describes the configuration you want to set up.

Areas you need to consider are:

- Network interface
- Addressing method
- Name service used, if any
- Dial in as well as dial-out support
- Routing requirements

### Remote Computer-to-Network Configuration

The remote computer-to-network is the most common asynchronous PPP configuration. Use it to configure machines in remote offices or user’s homes that will dial out over a point-to-point PPP link to a dial-in server on a network

- *Network interface* – This point-to-point link uses the `ipdptp`*n* virtual network interface. You will need to specify it in the configuration files of all remote machines that dial out to a network.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. For remote hosts, you probably should use existing host names and IP addresses. Refer to ”Determining IP Addressing for Your PPP Link” on page 123 for complete details.

- *Name service* – NIS and NIS+ name services are not recommended for remote hosts. These services generate a great deal of network traffic, often at unexpected times. The DNS name service is more efficient for this type of configuration. You may want to set up DNS, as described in *NIS+ and FNS Administration Guide*, on each remote host. If you don't use DNS, PPP will use the `/etc/inet/hosts` file on the remote machine.
- *Dial-in and dial-out support* – Remote hosts usually implement dial-out communications only. They do not allow other machines to dial in to them directly. Therefore, you will have to update the UUCP files on each to support dial-out communications, as explained in "Editing UUCP Databases" on page 136.
- *Routing requirements* – Because RIP is part of the Solaris TCP/IP protocol stack, it runs by default on remote hosts. You should turn off RIP to improve performance, if necessary, and instead use static routing. See "To Select Static Routing on a Host" on page 85 and "Turning Off RIP" on page 127 for details.

## *Remote Host-to-Remote Host Configuration*

Use the host-to-host configuration to establish point-to-point communications between two remote hosts in different physical locations. This configuration is useful for two standalone machines in remote offices that need to exchange information. No physical network is involved.

- *Network interface* – This basic point-to-point link uses the `ipdptp`*n* virtual network interface. You must specify the interface in the configuration files of both endpoints.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. Use the existing host names and the IP addresses assigned to the primary network interface, if they already exist. Otherwise, create IP addresses for the endpoints. Refer to "Determining IP Addressing for Your PPP Link" on page 123 for complete details.
- *Name service* – Because only two peer hosts are involved, you don't need a true name service. The `/etc/inet/hosts` files on both peer hosts are used for address resolution.

- *Dial-in and Dial-out support* – Both machines need to perform dial-in and dial-out operations. You will have to modify the UUCP databases and `/etc/passwd` on both endpoints.
- *Routing requirements* – Because RIP is part of the Solaris TCP/IP protocol stack, it runs by default on remote hosts. You should turn off RIP to improve performance, if necessary, and instead use static routing. See "To Select Static Routing on a Host" on page 85 and "Turning Off RIP" on page 127 for details.

## *Network-to-Network Configuration*

Use the network-to-network PPP configuration to create an internetwork joining two networks in physically separate locations. In this case, modems and PPP software function as the router connecting the networks.

- *Network interface* – The point-to-point link uses the `ipdptp`*n* virtual network interface. You must specify `ipdptp`*n* in the configuration files for both endpoint machines joining the two networks.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. Two possible addressing scenarios exist for this type of configuration; they are explained in "Determining IP Addressing for Your PPP Link" on page 123.
- *Name service* – NIS and NIS+ name services can function over this type of PPP link; however, each network should be a separate domain. If you use DNS, both networks can be part of a single domain. Refer to *NIS+ and FNS Administration Guide* for details. If you use local files for name service, the `/etc/inet/hosts` files on both endpoint machines are used for address resolution. They must contain the host names and IP addresses of every host on each network that will be allowed to communicate over the link.
- *Dial-in and Dial-out support* – Both network endpoint machines need to perform dial-in and dial-out operations, so you should update their UUCP and `/etc/passwd` files.
- *Routing requirements* – The endpoints in a network-to-network link usually run RIP in order to exchange routing information. Do not disable RIP for this configuration.

## *Dial-in Server With Dynamic Point-to-Point Links*

A dynamic point-to-point link is one of two types of configurations that you can use for a dial-in server functioning as the network endpoint that remote hosts will access. In this configuration scheme, the server connects to its remote hosts over a dynamically allocated point-to-point link. The dial-in server uses its dynamic links on an as-needed basis to establish communications with the remote hosts it serves.

- *Network interface* – The dynamic point-to-point link uses the `ipdptp*` virtual network interface with an asterisk wildcard character. The asterisk enables the link to be allocated dynamically. You must specify this interface in the configuration file.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. Refer to ”Determining IP Addressing for Your PPP Link” on page 123 for complete details.
- *Name service* – Although NIS and NIS+ are not recommended for remote hosts, the dial-in server in a remote host-to-network configuration can be an NIS client on the network to which it is physically connected. If NIS is on the server’s physical network, make sure that the NIS maps are updated with the host names and IP addresses of the remote hosts. You can use DNS on the dial-in server and its remote hosts. For more information regarding DNS and name services in general, refer to *NIS+ and FNS Administration Guide*. If you use local files for name service, PPP will use the `/etc/inet/hosts` file on the dial-in server for address resolution.
- *Dial-in support* – You must update the `/etc/passwd` file on the dynamic point-to-point dial-in server. The dynamic link server does not directly dial out to the remote hosts.
- *Routing requirements* – Because RIP is part of the Solaris TCP/IP protocol stack, it runs by default on remote hosts. You should turn off RIP to improve performance, if necessary, and instead use static routing. See ”To Select Static Routing on a Host” on page 85 and ”Turning Off RIP” on page 127 for details.

## Multipoint Dial-in Server

A multipoint link is one of two types of configurations that you can use for a dial-in server functioning as the network endpoint that remote machines will access. In this configuration scheme, the dial-in server connects to multiple remote hosts over the same multipoint link**.** The remote hosts always connect to the dial-in server over a point-to-point link, as explained on page 118.

Use this configuration when you want to define a separate network of remote hosts and their dial-in server.

- *Network interface* – The multipoint link uses the `ipd`*n* virtual network interface. You must specify this interface in the configuration file for the dial-in server.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. Refer to ”Determining IP Addressing for Your PPP Link” on page 123 for complete details. You must create a separate network for the machines on the multipoint link. See ”Assigning a Network Number to the PPP Link” on page 126 for more information.
- *Name service* – Although NIS and NIS+ are not recommended for remote hosts, the dial-in server in a remote host-to-network configuration can be an NIS client on the physical network to which it is connected. If NIS is on the server’s physical network, make sure that the NIS maps are updated with the host names and IP addresses of the remote hosts. You can use DNS on the dial-in server and its remote hosts. For more information regarding DNS and name services in general, refer to *NIS+ and FNS Administration Guide*. If you use local files for name service, PPP will use the `/etc/inet/hosts` file on the dial-in server for address resolution.
- *Dial-in and dial-out support* – The multipoint dial-in server functions as a network router between its PPP virtual network and the physical network to which it is connected. It dials out to its remote hosts whenever it receives IP traffic from the physical network destined for its PPP network. Therefore, you must configure the multipoint dial-in server for both dial-in and dial-out support, and update its UUCP and `/etc/passwd` files.
- *Routing requirements* – The `ipd`*n* interface does not support RIP; there is no need to disable it.

### Hosts on a Virtual Network

Use a virtual network configuration to connect three or more physically separated computers into a virtual network of phone lines, modems, and PPP software.

- *Network interface* – This type of configuration requires a multipoint link, which uses the `ipd`*n* virtual network interface. This interface connects each endpoint system with the other endpoints on the virtual network.
- *Addressing method* – The configuration file must include the host names or IP addresses of the machines that will communicate over the link. Refer to ”Determining IP Addressing for Your PPP Link” on page 123 for more information. You must assign a network number to the virtual network. Refer to ”Creating a Unique IP Address and Host Name” on page 125 for complete details.
- *Name Service* – You can run NIS and NIS+ for the virtual network; however, this may affect the performance of the link. DNS is a better alternative. Refer to *NIS+ and FNS Administration Guide* for instructions on setting up these name services. If you use files for name service, be sure to update `/etc/inet/hosts` on each machine with the host names and IP addresses of all machines comprising the virtual network.
- *Dial-in and dial-out support* – All machines in the virtual network must be configured for both dial-in and dial-out operations, so you should update their UUCP and `/etc/passwd` files.
- *Routing requirements* – The `ipd`*n* interface does not support RIP; you do not need to disable it.

## Determining IP Addressing for Your PPP Link

To enable communications over the PPP link, the machine at one end of the link must know the host name and IP address of the peer host on the other end of the link. The PPP configurations often require a particular addressing scheme. This section explains the addressing schemes and where each should be used.

### Specifying IP Addresses

On each endpoint machine, you specify addressing information in these places:

- `/etc/asppp.cf` configuration file
- `/etc/inet/hosts` file
- NIS+, NIS, or DNS databases, if applicable

When you edit the local machine's `asppp.cf` file, you must provide the host names and, in certain cases, the IP addresses for each endpoint machine to be on the link. For example, you must type either the IP addresses or host names for each endpoint as arguments in the `ifconfig` section in the configuration file:

```
ifconfig ipdptp0 plumb 192.99.44.01 192.99.44.02 up
```

See Chapter 9, "Configuring PPP" for information regarding the format of `/etc/asppp.cf`.

Additionally, to enable communications, you must add the IP address and host name of the remote endpoints to the `hosts` database on the local end point by editing `/etc/inet/hosts`. This process is explained in "Configuring Network Clients" on page 75.

## *Types of Addressing Schemes*

You have a choice of several addressing schemes for PPP, depending on your configuration type. Before you edit the `asppp.cf` file and `hosts` database, you must decide on the appropriate addressing scheme for your configuration. These schemes include

- Using the same IP addresses for the PPP endpoints as is assigned to their primary network interface in their local `/etc/inet/hosts` files
- Assigning a unique IP address for each PPP endpoint
- Assigning a new network number for the network created by the PPP link

### *Using the Same IP Address as the Primary Network Interface*

This addressing scheme is appropriate for point-to-point links only. In this scheme, you specify the addresses of the primary network interface for each endpoint. (See Chapter 1, "Overview of Network Administration," for more information about the primary network interface.) These endpoints might be

- Two standalone machines communicating over the PPP link (if they have existing IP addresses)
- Two network endpoints communicating over the PPP link
- Remote host connecting to a network dial-in server through a point-to-point link
- Dial-in server connecting to remote hosts through a dynamically allocated point-to-point link

When you edit the `/etc/inet/hosts` file on a local endpoint, supply the IP address of its primary network interface and host name and the IP address of the peer host on the other end of the link.

### *Creating a Unique IP Address and Host Name*

In this method, you assign a unique host name and IP address to the PPP network interface. (You might want to call the interface *hostname*-ppp.) Use this addressing scheme for

- Endpoint machines on a network used as a multipoint dial-in server.
- Machines on a virtual network.
- Remote host that uses a dedicated IP address for communicating with a dial-in server over a dynamically allocated PPP link. (Note that this is not a requirement for the dynamic link configuration.)
- Machine that is also configured as a router for a physical network, such as Ethernet or Token Ring.
- Machine in a standalone-to-standalone configuration that does not have an existing IP address. (The PPP interface becomes the primary network interface.)

You must specify the unique address and host name for the PPP network interface in the `asppp.cf` configuration file.

To create the new host name and IP address, simply add it to the `/etc/inet/hosts` file on the endpoint machines, as described in ”hosts Database” on page 53.

### *Assigning a Network Number to the PPP Link*

You create a new network number for the PPP configuration when it involves

- Virtual networks of computers communicating through PPP multipoint links (required)
- A multipoint dial-in server and its remote hosts (required)
- The PPP link between two networks, particularly when one or both of the network endpoint machines are also routers for a physical network (optional)

(See Chapter 3, “Planning Your Network,” for information on network numbers.)

The PPP link becomes a *virtual network*, since it does not involve any physical network media. You need to type its network number in the `networks` database on all endpoint machines, along with the network numbers of the networks being linked.

Here is a sample `/etc/inet/networks` file for an internetwork with PPP:

*Code Example 8-1* `/etc/inet/networks` File for an Internetwork With PPP

```
kalahari     192.9.253
negev        192.9.201
nubian-ppp   192.29.15
```

In the sample file, `kalahari` and `negev` are two local area networks, and `nubian-ppp` is the name of the PPP link.

## *Routing Considerations*

The RIP routing protocol runs on Solaris TCP/IP networks by default. In most cases, you should leave RIP running on point-to-point links. However, if you are having performance problems with the link, you may want to disable RIP on the point-to-point link.

**Note –** RIP will not be started on multipoint links. Therefore, you must set up static routing for the multipoint link. Refer to ”To Select Static Routing on a Host” on page 85 for instructions.

### *Turning Off RIP*

You can disable RIP on a point-to-point link through the file `/etc/gateways`. This file does not come with your operating system: you must create it with a text editor.

To turn off RIP, `/etc/gateways` must have the following entry:

```
norip ipdptpn
```

where `ipdptp`*n* represents the device name of the point-to-point PPP interface used.

For more information, refer to the `in.routed`(1M) man page.

## *PPP Hardware Requirements*

The basic PPP configuration involves a computer, a modem, and RS-232 telephone lines. However, before you configure, you need to verify whether the hardware you selected can support PPP. This section describes the hardware requirements for PPP.

- *Modem requirements* – To run PPP, each endpoint machine must have a modem that supports at least 9600 bps or faster bidirectional connections. Such a modem would implement the V.32 or V.32bis specification.
- *Serial port selection (for dial-in servers only)* **–** You can configure either serial port A or serial port B on most CPUs for PPP usage. Use the Solaris Serial Port Manager to initialize the ports on the dial-in server. *System Administration Guide, Volume II* contains instructions for selecting the appropriate port. If you have additional serial cards installed, you can also use their serial ports for PPP connections
- *Disk space* – You must have 300 Kbytes of free space in `/usr` to install PPP.

# 8

## File Space Requirements

You will need sufficient space in the following directories for the PPP software:

- `/usr`
- `/usr/kernel/drv`
- `/usr/kernel/strmod`
- `/usr/sbin`

PPP occupies approximately 243 Kbytes in `/usr` and 4 Kbytes in `/` (root).

### Checklist for Configuring PPP

Use this checklist to prepare for configuring PPP. It lists the information you will need to gather and the tasks you need to do before starting the configuration process.

1. Do you have 300 Kbytes of free space available in `/usr`? _____
2. Do you have 4 Kbytes of free space available in `/` (root)? _____
3. Do the modems for each endpoint support V.32 or V.32bis or higher? _____
4. Have you used the Serial Port Manager on the dial-in server to designate the serial port for the modem? _____
5. Have you ensured that Solaris PPP is installed on each endpoint machine? (If PPP hasn't been installed, you can use the `pkgadd` program or `admintool` software manager to install it.
   Refer to *SPARC: Installing Solaris Software* or *x86: Installing Solaris Software* for instructions.) _____
6. Have you ensured that there are no other versions of PPP running on each endpoint. (If there are, disable them, as explained in their documentation.) _____

7. Have you determined which IP addresses to use for all computers involved in the PPP link? _____
   List the host names and IP addresses of these machines here. _____
   _____
   _____
   _____
8. Write the name and IP address of the dial-in server (if applicable). _____
9. Write the name of the network interface that you need to use. _____

# 8

# Configuring PPP 9

This chapter contains procedures and information for configuring PPP. The example used in the text is for the configuration with both types of PPP links—remote hosts and their multipoint dial-in server. Chapter 11, “Tailoring Your PPP Link,” contains information for setting up other PPP configuration types.


| | |
|---|---|
| *Installing the PPP Software* | *page 132* |
| *Sample Network of Remote Hosts and Multipoint Dial-in Server* | *page 134* |
| *Editing UUCP Databases* | *page 136* |
| *Modifying the /etc/passwd File* | *page 138* |
| *Parts of Basic Configuration File* | *page 140* |
| *Configuration File for Multipoint Dial-in Server* | *page 142* |
| *Starting up and Stopping Your New PPP Link* | *page 146* |


## Overview of the Configuration Process

You have completed the preinstallation activities noted in Chapter 8, “Preparing Your PPP Configuration.” Now you can begin PPP configuration.

PPP requires

1. Installing the PPP software, if it isn’t already installed.
2. Editing the `/etc/inet/hosts` files on all machines involved.
3. Edit the UUCP database files for all dial-out machines.

4. Editing the `/etc/passwd` and `/etc/shadow` files for the dial-in machine.
5. Editing the `/etc/asppp.cf` file on each machine on the link.
6. Starting the link manager `aspppd` on each machine on a link.
7. Verifying that PPP is running successfully.

Although you don't have to perform Tasks 1–4 in order, you must complete them before you can edit the PPP-configuration file.

The sections in this chapter explain the procedures for configuring PPP.

## *Installing the PPP Software*

The PPP software is automatically included when you run the Solaris installation program and select the entire distribution. If you did not select the entire distribution, you need to install PPP as a separate package.

### *Verifying Installation*

Before proceeding further, you must check that the Solaris version of PPP is installed on all machines to be involved in the PPP link. On each endpoint involved in the link, type:

```
# pkginfo | grep ppp
```

If PPP is installed, the following package names will be displayed:

```
SUNWpppk       # Contains kernel modules
SUNWapppu      # Contains the link manager and login service
SUNWapppr      # Contains configuration files
```

If PPP is not installed on an endpoint system, install it using either the `pkgadd` program or `admintool` software manager.

---

**Note –** When using `pkgadd` to install PPP, you must install the packages in the order listed in the preceding screen box.

---

Refer to *System Administration Guide, Volume I* for more information about `pkgadd` and `admintool` software manager.

## *Sample PPP Configuration*

This and the following sections show you how to edit the appropriate files to support the most common PPP configuration: remote hosts and their dial-in server. Figure 9-1 on page 134 illustrates the configuration used as the example for this chapter. It depicts three remote machines (`nomada`, `nomadb`, `nomadc`) and their dial-in server `nubian`, which compose the network 192.41.43. This is a separate network from the local area network 192.41.40, to which dial-in server `nubian` is directly attached. Network 192.41.40 runs NIS as its name service.

The IP number shown for each remote host is the address of its PPP network interface. However, the dial-in server has a specially created IP address for the PPP interface, 192.41.43.10, in addition to the IP address for its primary network interface, 192.41.40.45.

# 9

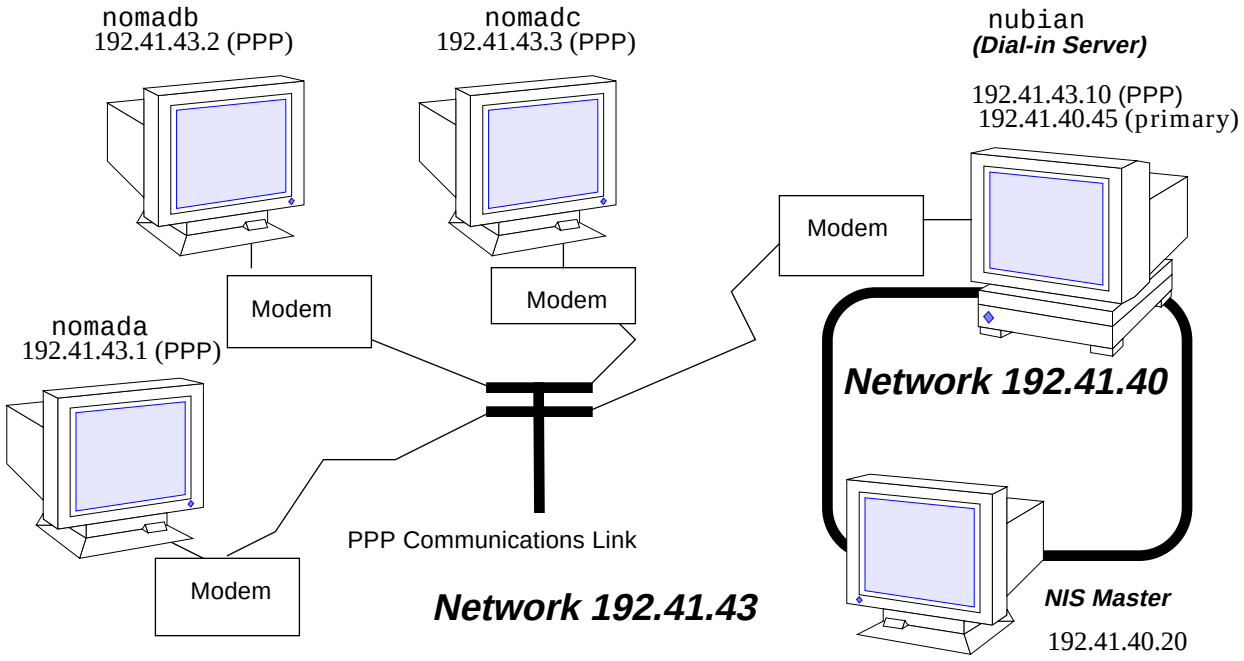



*Figure 9-1* Sample Network of Remote Hosts and Multipoint Dial-in Server

## *Editing the* `/etc/inet/hosts` *File*

After ensuring that PPP is installed on every machine involved in your configuration, your next task is to edit the `/etc/inet/hosts` files on each machine. You must add host information to the `hosts` database for every machine on the other end of the PPP link that the local machine needs to communicate with.

---

**Note –** You must update `/etc/inet/hosts` regardless of the name service in use on the physical network. This is necessary because PPP starts before the name service daemons during the booting process.

---

### ▼ How to Configure the Remote Machine's `hosts` Database

1. **Become superuser and prepare to edit the `/etc/inet/hosts` file.**
2. **Add an entry with the IP address and host name of the PPP network interface for the dial-in server on the other end of the link.**
   In Figure 9-1 on page 134, `nomada` will have in its `/etc/inet/hosts` file an entry with the IP address for dial-in server `nubian`'s PPP network interface. This would be true also for the `/etc/inet/hosts` files for `nomadb` and `nomadc`.
3. **Add entries with the IP addresses of any machines on the dial-in server's physical network that the remote host can remotely log in to.**
   The `/etc/inet/hosts` file on `nomadc` would look like:

```
# Internet host table
#
127.0.0.1          localhost      loghost
192.41.43.3        nomadc
192.41.43.10       nubian-ppp
192.41.40.20       nismaster
```

4. **Update the databases on the name server (if the network has one) with the host names and IP addresses of the remote hosts.**

## *Multipoint Dial-in Server* `hosts` *Database*

Multipoint dial-in servers must have a unique IP address for the PPP interface, besides the local IP address for the primary network interface. When configuring the `hosts` database for the dial-in server, you need to do the following procedure.

### ▼ How to Configure the Dial-In Server's `hosts` Database

1. **Add an entry with the IP address for the PPP interface to the `/etc/inet/hosts` file for the dial-in server.**
   For example, the `/etc/hosts` file on dial-in server `nubian` in Figure 9-1 on page 134 would have these entries.

```
# Internet host table
#
127.0.0.1           localhost      loghost
192.41.43.10        nubian-ppp
192.41.40.45        nubian
```

2. **For configurations where the server's physical network does not use a name service:**
   a. **Add entries to the server's `/etc/inet/hosts` files for each remote host served.**
   b. **Add entries for the remote hosts to the `/etc/inet/hosts` files of every machine on the physical network permitted to communicate with the remote machines.**
3. **Add a new network number to the dial-in server's `/etc/inet/networks` file for the network comprised of the server and its remote hosts.**
   Refer to Section , "Assigning a Network Number to the PPP Link," on page 126 for more information.

## *Editing UUCP Databases*

Before a machine can dial out over the PPP link, you must edit these files in its UUCP database:

- `/etc/uucp/Devices`
- `/etc/uucp/Dialers`
- `/etc/uucp/Systems`

You must edit these files for remote hosts serving as PPP dial-out machines. Additionally, you must edit these files on the dial-in server if it is to dial out to the remote hosts (a requirement for multipoint dial-in servers). Chapter 12, "UUCP Databases and Programs," describes these files in detail.

## *Updating* /etc/uucp/Devices *for PPP*

The /etc/uucp/Devices file must contain entries for every communications device that a particular host uses or must know about. For example, if a machine uses a US Robotics V.32bis modem as part of the PPP link, you should ensure that /etc/uucp/Devices has an entry similar to the following:

```
# Use these if you have a USrobotics V.32bis modem on Port B.
ACUEC   cua/b - 9600 usrv32bis-ec
ACUEC   cua/b - 19200 usrv32bis-ec
ACUEC   cua/b - 38400 usrv32bis-ec
```

Be sure that the Devices file on each PPP endpoint machine has an entry describing its modem. For more information about /etc/uucp/Devices, refer to "/etc/uucp/Devices File" on page 198.

## *Updating* /etc/uucp/Dialers *for PPP*

The /etc/uucp/Dialers file must have an entry describing the conversation with the modem attached to your PPP endpoint machine. Here is a sample entry for a US Robotics V.32bis modem that will be part of a PPP link:

```
usrv32bis-ec =,-,  "" \dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&&B1W\r\c OK\r \EATDT\T\r\c
CONNECT\s14400/ARQ STTY=crtscts
```

The first parameter in the entry, usrv32bis, corresponds to the last parameter in the /etc/uucp/Devices file, linking them together. The remainder of the entry describes the characters that the modem will send, those that it expects to receive, and so on. Table 12-16 on page 212 defines the control codes used in the Dialers file.

Be sure that an entry is in the Dialers file for the modem attached to each dial-out endpoint on your link. If you are unsure of the correct conversation for a particular modem, refer to *System Administration Guide, Volume II* and the operating manual for the modem.

## *Updating* /etc/uucp/Systems *for PPP*

The /etc/uucp/Systems file contains entries for every machine to which the local host can dial out. Information in an entry might include the remote host's phone number, the line speed, and so on. Here is an example that host nomadb in Figure 9-1 on page 134 might have for its dial-in server:.

```
nubian-ppp  Any ACUEC 38400 5551212 "" P_ZERO "" \r\n\c login:-\r\n\c-login:-\r\n\c-login:-
EOT-login: bnomad password: Secret-Password
```

The first field gives the server's host name, nubian-ppp, a value used by the asppp.cf file keyword peer_system_name. ACUEC and 38400 refer to the device and speed, and are used to select an entry from the /etc/uucp/Devices file. The remaining information includes the phone number of the machine that nomadb wants to dial in to, the login name that nomadb will use to log in, and so on. "/etc/uucp/Systems File" on page 191 fully defines the parameters you need to supply to the Systems file.

On each remote host in your configuration, you must add an entry for its dial-in server. Note that you can have additional entries in the /etc/uucp/Systems file for other machines to which the host will dial out for UUCP communications and for other PPP dial-in servers.

If the dial-in server will also directly dial out to remote hosts, you must add entries to its Systems file describing each of these remote hosts.

## *Modifying the* /etc/passwd *File*

To configure a dial-in server, you must also edit the /etc/passwd and /etc/shadow files.

You must add entries to the /etc/passwd file on the dial-in server for each user on a remote host authorized to log in to the server. When a remote host calls the dial-in server, it reads its UUCP databases and passes the server a user name or user ID for the host initiating the call. The server then verifies this user information in its /etc/passwd file.

If the user's password is authenticated, the server then logs the user in to a special shell for PPP hosts, `/usr/sbin/aspppls`. The server gets this information from the login shell entry in its `/etc/passwd` file. Using the example in Figure 9-1 on page 134, dial-in server `nubian` might have the following entries in its `/etc/passwd` file:

```
bin:x:2:2::/bin:
sys:x:3:3::/bin:
uucp:x:5:5::/usr/lib/uucp:
nuucp:x:9:9::/var/spool/uucppublic:/usr/lib/uucp/uucico
news:x:6:6::/var/spool/news:/bin/csh
sundiag:x:0:1:System Diagnostic:/usr/diag/sundiag:/usr/diag/sundiag/sundiag
lily:x:20:99:Dial-in Operator:/home/nubian/lily:/bin/csh
nomada:x:21:99:R. Burton:/:/usr/sbin/aspppls
nomadb:x:22:99:T. Sherpa:/:/usr/sbin/aspppls
nomadc:x:23:99:S. Scarlett:/:/usr/sbin/aspppls
```

Refer to *System Administration Guide, Volume II* for information about the `/etc/passwd` file.

**Note –** In addition to the information in the `/etc/passwd` file, you update the `/etc/shadow` file with the passwords for the login names used by each endpoint machine permitted to dial in to the server. For more information, refer to *System Administration Guide, Volume II.*

## *Editing the* `/etc/asppp.cf` *Configuration File*

The `/etc/asppp.cf` configuration file provides the PPP link manager on one endpoint machine with information about the machine on the other end of the link—or the machines on the other end of a multipoint (or dynamic point-to-point) link. When the machine boots, the link manager uses this information to establish and maintain communication with a remote endpoint.

## *Parts of Basic Configuration File*

The basic `asppp.cf` configuration file must contain at least two main sections: an `ifconfig` line and at least one `path` section. It can also contain a `defaults` section, which you use when you want to set the default values for an endpoint. (Refer to Chapter 11, "Tailoring Your PPP Link," for a description of keywords used in the `defaults` section.)

Code Example 9-1 shows a basic configuration file such as you would create for a remote host to establish a point-to-point link with a dial-in server.

*Code Example 9-1* Basic Configuration File

```
ifconfig ipdptp0 plumb nomada nubian-ppp up
   path
      interface ipdptp0
      peer_system_name nubian-ppp     # The name in the /etc/uucp/Systems file
      inactivity_timeout 300           # Allow five minutes before timing out
```

### `ifconfig` *Section of the* `asppp.cf` *File*

The `asppp.cf` file must contain an `ifconfig` section with this syntax:

`ifconfig` *interface-number* `plumb` *local-machine remote-machine* `up`

Here is a description of the fields:

- `ifconfig` – Tells the link manager to run the `ifconfig` command and begin configuring the PPP interface.
- *interface-number* – Identifies the PPP interface `ipdptp`*n* for a point-to-point link or `ipd`*n* for a multipoint link. (Replace the *n* with the number of the interface.)
- `plumb` – Option of `ifconfig` that enables IP to recognize the interface.
- *local-machine* – Gives the name of the local endpoint, which can be the local host name or IP address.
- *remote-machine* – Gives the name of the remote endpoint, which can be the remote host name or IP address.

- `up` – Option of `ifconfig` that marks the interface just described as up.

The link manager first runs the `ifconfig` command on the local machine to configure the `ipdptp0` point-to-point interface. The zero in `ipdptp0` gives the device number of the interface. The `plumb` option performs various activities necessary for IP to recognize the `ipdptp0` interface. `nomada` is the name of the local host. `nubian-ppp` is the name of the dial-in server to which `nomada` will connect through the point-to-point link. The `ifconfig` option `up` marks the `ipdptp0` interface as up.

---

**Note –** For more information about `ifconfig`, see Chapter 10, "Troubleshooting PPP" and the `ifconfig`(1M) man page.

---

## `path` *Section of the* `asppp.cf` *File*

The `path` section of the configuration file tells the link manager the name of the remote endpoint and the name of the interface linking the endpoint machines. At a minimum the `path` section should contain the following lines:

```
path
 interface interface-number
  peer_system_name endpoint-name
```

### `interface` ***Keyword***

This keyword defines the PPP interface (either `ipdptp`*n* or `ipd`*n*). In Code Example 9-1 on page 140, the following information appears in the `path` section:

```
      interface ipdptp0
      peer_system_name nubian-ppp
```

The `interface` keyword identifies `ipdptp0` as the point-to-point interface that local endpoint `nomada` will use to communicate with the remote endpoint in the manner described in this `path` section. It associates the `peer_system_name` with the interface.

### `peer_system_name` *Keyword*

On a dial-out machine such as a remote host, the `peer_system_name` keyword takes the host name of the remote endpoint as its argument. This is the name of the remote endpoint given in `/etc/uucp/Systems`. The name need not be the same as the host name on the corresponding `ifconfig` line.

---

**Note –** The argument to the `peer_system_name` keyword for a dial-in server has a different value. See "Configuration File for Multipoint Dial-in Server" on page 142 for details.

---

In the example on Code Example 9-1 on page 140, `peer_system_name` identifies dial-in server `nubian-ppp` as the remote endpoint at the other end of this link. When the link manager reads the `asppp.cf` file, it then looks for the entry for `nubian-ppp` in the `/etc/uucp/Systems` file. (Recall that the `Systems` file contains information about how to set up communications with the remote endpoint, including that machine's telephone number. Refer to "Updating /etc/uucp/Systems for PPP" on page 138.)

### `inactivity_timeout` *Keyword*

The `inactivity_timeout` keyword is optional. It tells the link manager that the link can remain inactive for the interval designated. When that interval is passed, the link manager knows to automatically disconnect the link. The default interval is two minutes; you do not have to use `inactivity_timeout` unless you require a different inactivity interval.

### *Additional Keywords*

You can supply other keywords in the `asppp.cf` file to define how endpoint machines should communicate. Chapter 11, "Tailoring Your PPP Link," has complete information about these keywords.

## *Configuration File for Multipoint Dial-in Server*

The `asppp.cf` configuration file for a multipoint dial-in server contains the same basic sections as that for a point-to-point link: an `ifconfig` section, at least one `path` section, and, if desired, a `defaults` section.

Code Example 9-2 shows a configuration file for the dial-in server `nubian` introduced in Figure 9-1 on page 134.

*Code Example 9-2* Configuration File for a Multipoint Dial-in Server

```
ifconfig ipd0 plumb nubian-ppp up

path
    interface ipd0
    peer_system_name tamerlane  # The user name this remote
                                # machine logs in with when it
                                # dials this server
    peer_ip_address nomada
                               # nomada is a remote machine that
                                # dials in to this server

# nomadb is another remote machine that dials in to nubian

path
    interface ipd0
    peer_system_name lawrence
    peer_ip_address nomadb

# nomadc is another remote machine that dials in to nubian

path
    interface ipd0
    peer_system_name azziz
    peer_ip_address nomadc
```

## `ifconfig` *Section for Multipoint Dial-in Server*

The `ifconfig` section for a multipoint dial-in server has a slightly different syntax than that for a point-to-point link. This syntax is:

`ifconfig ipd`*n* `plumb` *server-name* `up`

The major difference is that no destination end points are listed as arguments to `ifconfig`. Instead, the link manager picks up this information from the `path` section of the `asppp.cf` file.

In Code Example 9-2 on page 143, the link manager first runs the `ifconfig` command on the dial-in server to configure multipoint interface `ipd0`. The zero in `ipd0` gives the device number of the interface. The option `plumb` performs various activities necessary for IP to recognize the `ipd0` interface. The `ifconfig` option `up` marks interface `ipd0` as up.

**Note –** You may have to supply a netmask + parameter on the `ifconfig` line if you use subnetting.

## `path` *Section for Multipoint Dial-in Server*

The `path` section of the `asppp.cf` file tells the link manager the name of the remote endpoint and the name of the interface linking the endpoint machines. However, on a multipoint dial-in server, you can include more than one `path` section. Additionally, some of the arguments to the keywords are used differently with multipoint links.

```
path
        interface interface-number
        peer_system_name endpoint-username
        peer_ip_address endpoint-hostname
```

You need to define a `path` section for each nomadic endpoint with which the dial-in server will establish connections.

### `interface` ***Keyword***

For a multipoint dial-in server, the `interface` keyword defines the PPP interface `ipd`*n*. You must specify the same PPP interface in the `path` section for every endpoint that will communicate with the server through this interface.

### `peer_system_name` ***Keyword***

The `peer_system_name` keyword takes a slightly different argument for a dial-in machine than a dial-out machine. For a dial-in server, this argument is the login name used by the remote host when it tries to establish communications with the server. This user name must already be present in the

server's /etc/passwd file. When the login service reads this name, it verifies the user name in the /etc/passwd and /etc/shadow files enabling communications.

In this excerpt from Code Example 9-2:

```
path
    interface ipd0
    peer_system_name scarlett
    peer_ip_address nomadc
```

the argument to peer_system_name is scarlett, indicating that when nomadc logs in to nubian-ppp, it uses the login name scarlett.

#### peer_ip_address *Keyword*

The peer_ip_address keyword is required for multipoint links. It takes the host name or IP address of the remote endpoint as its argument. The example above uses the host name nomads as the argument to keyword peer_ip_address.

#### *Additional Keywords*

You can supply other keywords in the asppp.cf file to define how endpoint machines should communicate. Refer to Chapter 11, "Tailoring Your PPP Link," for a complete list of keywords.

## *Editing the Configuration File*

When editing asppp.cf:

- Separate keywords in the configuration file by white space (blanks, tabs, and newlines).
- Use a # sign before all character strings meant as comments. All characters placed between a # sign and the next new line are considered comments and ignored.

There are no other format requirements for the placement of the keywords in the file.

### ▼ How to Edit the `asppp.cf` Configuration File

1. **Become superuser on one endpoint machine and change to the `/etc` directory.**
2. **Edit the generic `asppp.cf` file to add the information defining the PPP link for this machine.**
3. **Save the file, making sure the permissions are set to `600`.**
4. **Change to the `/etc` directories on the remaining endpoints and repeat Steps 2 and 3.**

## *Adding PPP Security*

Once you have completed installing PPP on every machine involved in your configuration, you can add either PAP or CHAP levels of security for the PPP link by modifying the `asppp.cf` file. Refer to “Editing asppp.cf for PAP/CHAP Security” on page 176.”

## *Starting up and Stopping Your New PPP Link*

You can start PPP either automatically, at boot time, or manually from the command line.

### ▼ How to Manually Start PPP

You can start PPP manually, although this is not normally required.

1. **Become superuser and type:**

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

### ▼ How to Verify That PPP Is Running

1. **Run the *ps* command:**

```
# ps -e | grep asppp
```

The resulting output from `grep` should list the `aspppd` daemon, indicating that PPP is running.

2. **If you do get results, verify that you can reach the remote PPP link by typing:**

```
# ping remote-host 300
```

This version of `ping` sets a timeout value of 5 minutes (300 seconds). You should receive output similar to *remote-host* `is alive`. If you receive a different notice, such as *remote-host* `unreachable`, route configuration has failed.

3. **Check for errors in the configuration process by examining the log file.**

```
# tail /var/adm/log/asppp.log
```

The `asppp.log` will contain error messages if any errors were encountered during configuration.

Go to Chapter 10, “Troubleshooting PPP,” for information on troubleshooting and problem solving.

## ▼ How to Stop PPP

1. **To stop PPP operations on your network, type:**

```
# /etc/init.d/asppp stop
```

# 9

# Troubleshooting PPP 10

This chapter is organized as a series of checks to make after you have configured PPP on your network. Thereafter, whenever you have trouble communicating over the PPP link, you can use PPP diagnostics to help troubleshoot problems.


| | |
|---|---|
| *Checking Hardware* | *page 150* |
| *Checking Interface Status* | *page 150* |
| *Checking Connectivity* | *page 151* |
| *Checking Interface Activity* | *page 151* |
| *Checking the Local Routing Tables* | *page 152* |
| *Checking Permissions* | *page 153* |
| *Checking Packet Flow* | *page 153* |
| *Using PPP Diagnostics for Troubleshooting* | *page 155* |


In summary, you should do these checks in the following order:

1. Hardware
2. Interface status
3. Connectivity
4. Network interface activity
5. Local routing tables

6. Permissions

7. Checking packet flow

If PPP passes all the tests, you should be able to use TCP and UDP services such as `rlogin`, `telnet`, and `ftp` over the link. If the link still fails, turn on PPP diagnostics for assistance in troubleshooting.

The next sections describe these checks in detail.

## Checking Hardware

Make sure that all modem and power cables are tightly seated. If you are having problems with PPP, always check the modems, cables, serial card, and phone lines first.

## Checking Interface Status

After PPP is started, you can use `ifconfig` to monitor the current state of the line, using only the PPP interface name as an argument. Following are sample outputs from `ifconfig` for PPP links that are running.

*Code Example 10-1* `ifconfig` Output for Point-to-Point Link

```
nomadb# ifconfig ipdptp0

ipdptp0: flags=28d1<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST,UNNUMBERED> mtu 1500
        inet 129.144.111.26 --> 129.144.116.157 netmask ffff0000
        ether 0:0:0:0:0:0
```

You will receive output similar to that in Code Example 10-1 for both standard and dynamic point-to-point links.

*Code Example 10-2* `ifconfig` Output for Multipoint Link

```
nubian# ifconfig ipd0

ipd0: flags=c1<UP,RUNNING,NOARP> mtu 1500
        inet 129.144.201.191 netmask ffffff00
        ether 0:0:0:0:0:0
```

If `ifconfig` does not display `UP` and `RUNNING`, you did not configure PPP correctly. For more information on `ifconfig`, see ”ifconfig Command” on page 92 and the `ifconfig`(1M) man page.

## *Checking Connectivity*

Use the `ping` command to verify that the connection is up or can be established. For example, consider the following simple round-trip test:

```
# ping elvis
```

where `elvis` is the name of the PPP interface on the remote host. If the resulting display is

```
elvis is alive
```

then packets can be sent to and received from `elvis`. If not, a routing problem exists at some point between the local and remote hosts. For more information on `ping`, refer to ”ping Command” on page 90 and the `ping`(1M) man page.

## *Checking Interface Activity*

Use the `netstat` command as follows to check that packets are being sent and received correctly:

```
# netstat –i
```

Refer to ”netstat Command” on page 93 and the `netstat(1M)` man page.

## *Checking the Local Routing Tables*

Use the `netstat` command to display the local routing tables:

```
# netstat -r
```

The following is sample output:

```
Routing tables

Destination   Gateway    Flags  Refcnt Use      Interface

sahara        deserted   UGH    0      0        ie1
karakum       labia      UGH    0      0        ie1
frodo         bilbo      UGH    1      12897    ipdptp0
route7        route7     UGH    0      0        ie0
eastgate      route71    UGH    0      158      ie0
backbone      pitstopbb  U       1     16087     ie1
dresdenpc     route1     UG     0      0        ie1
loopback      localhost  U      2      113436   lo0
swan-bb       pitstop    U      4063   146044   ie0
dallas2       route7     UG     0      0        ie0
trainingpc    route62    UG     0      0         ie1
```

Make sure there is a routing table entry for each possible destination network. In particular, PPP devices, listed under `Interface`, should be matched with the appropriate host names listed under `Gateway`. The `Gateway` entry should, in turn, be matched with the correct entry under `Destination`.

Otherwise, if you are using *static routing*, add the appropriate static routes. If you are using *dynamic routing* with `in.routed`:

1. **Verify that** `in.routed` **is running by typing:**

```
# ps -e | grep route
```

   If the routing tables still don't look correct, become superuser and continue with the next steps.

2. **Kill** `in.routed` **by typing the process ID you got from** `ps -e` **as the argument to** `kill`**. For example, if 1384 was the process ID, you would type:**

```
# kill 1384
```

3. **Flush the routing tables as follows:**

```
# /usr/sbin/route -f
```

4. **Restart** `in.routed`**:**

```
#/usr/sbin/in.routed
```

## *Checking Permissions*

If you attempt to use `rsh` and receive the message `Permission denied`, the remote system's `/etc/hosts.equiv` or `/.rhosts` file does not contain the sending system's host name or does not contain the line `+`.

## *Checking Packet Flow*

Check the packet flow next. Use the `snoop` command to observe packets from the network and observe their contents.

*Code Example 10-3* Sample Output from snoop

```
# snoop -d ipdptp0
Using device ipdptp0 (promiscuous mode)
corey -> pacifica7    RLOGIN C port=1019
       hugo -> ponc3          RPC R XID=22456455 Success
       ponc3 -> hugo          NFS C WRITE FH=1B29 at 32768

    commlab3 -> commlab4      TELNET R port=34148
    commlab4 -> commlab3      IP  D=129.144.88.3 S=129.144.88.4 LEN=46, ID=41925
    commlab3 -> commlab4      TELNET R port=34148
    commlab4 -> commlab3      ICMP Echo request
    commlab3 -> commlab4      ICMP Echo reply
    commlab4 -> commlab3      FTP C port=34149
    commlab4 -> commlab3      FTP C port=34149
    commlab3 -> commlab4      FTP R port=34149
    commlab4 -> commlab3      FTP C port=34149
```

The `ipdptp0` device name mentioned in the first line of the output "`Using device ipdptp0`" indicates a point-to-point connection.

---

**Note –** You need to have the link up and some traffic generated in order to use `snoop` to check the line status.

---

`snoop` captures packets from the network and displays their contents. It uses both the network packet filter and streams buffer modules to provide efficient capture of packets from the network. Captured packets can be displayed as they are received or saved to a file for later viewing.

`snoop` can display packets in a single-line summary form or in verbose multiline forms. In summary form, only the data pertaining to the highest-level protocol is displayed. For example, an NFS packet will have only NFS information displayed. The underlying RPC, UDP, IP, and Ethernet frame information is suppressed but can be displayed if either of the verbose options is chosen.

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

## *Using PPP Diagnostics for Troubleshooting*

If you have problems with a link after successfully establishing modem connections, you can use PPP-level diagnostics for troubleshooting. PPP-level diagnostics report detailed information about the activities of a link to help you determine where it is failing.

To obtain diagnostic information, add the line `debug_level 8` to the `path` section of the `asppp.cf` file. (If you are very knowledgeable about data communications, you may want to use debug level 9, which provides very detailed information.) Here is a sample configuration file that invokes PPP diagnostics.

```
ifconfig ipdptp0 plumb nomada nubian-ppp up
path
   interface ipdptp0
   peer_system_name nubian-ppp   #The name in the /etc/uucp/Systems file
   inactivity_timeout 300        #Allow five minutes before timing out
   debug_level 8                 #Start up PPP diagnostics for this link
```

For complete details about the `aspppd.conf` file, refer to ”Editing the /etc/asppp.cf Configuration File” on page 139.

### ▼ How to Set Diagnostics for Your Machine

Set diagnostics on the host you want to monitor as follows:

1. **Become superuser and change to the `/etc` directory.**
2. **Edit the current `asppp.cf` file and add the following to the `path` section:**

   ```
   debug_level 8
   ```

3. **Save the file, making sure the permissions are set to `600`.**

**4. Kill the current `aspppd` daemon and restart it.**

```
#kill -HUP PID
#aspppd
```

where *PID* is the process ID for `aspppd`.

PPP will now report diagnostic information in `/var/adm/log/asppp.log`.

## Analyzing Diagnostic Output

When a PPP link runs correctly, the `asppp.log` file includes diagnostic information in addition to its normal output. This section explains what the diagnostic messages mean. If your output is different, refer to RFC 1331.

### Host and Modem Setup

This section contains messages that occur as the local host sends configuration information to the modem, and then as the modem tries to dial the remote host. These initial activities are actually handled by the UUCP daemon. You might think of them as the UUCP portion of asynchronous PPP communications. (Refer to Chapter 12, “UUCP Databases and Programs,” for complete details on UUCP.)

The two messages below should always appear at the beginning of the session. They indicate that the `aspppd` daemon has started successfully.

```
11:53:33 Link manager (1057) started 04/14/94
11:53:33 parse_config_file: Successful configuration
```

The next line indicates that a packet was routed to the `ipdptp0` interface on the local host. It helps you to determine if a dial-out is occurring correctly. For example, if you tried to `ping` the remote machine and this message isn’t in `asppp.log`, the packet was lost, probably due to a routing problem.

Next, UUCP looks for an entry that matches `Ppac7` in a chat script in the `/etc/uucp/Systems` file. It then reports that it found an entry that had a device type ACUTEC. (For more information on the `Systems` file, refer to "/etc/uucp/Systems File" on page 191.)

```
11:53:46 process_ipd_msg: ipdptp0 needs connection
conn(Ppac7)
Trying entry from '/etc/uucp/Systems' - device type ACUTEC.
```

UUCP then finds the dialing information for an ACUTEC dialer in the `/etc/uucp/Devices` file. When it find the information, it opens the appropriate serial port on the local host and sets it with a speed of 9600. (For more information on `/etc/uucp/Devices`, see "/etc/uucp/Devices File" on page 198.)

```
Device Type ACUTEC wanted
Trying device entry 'cua/a' from '/etc/uucp/Devices'.
processdev: calling setdevcfg(ppp, ACUTEC)
fd_mklock: ok
fixline(8, 9600)
gdial(tb9600-ec) calle
```

UUCP looks for the entry `tb9600` in the `/etc/uucp/Dialers` file and then sends out these messages.

```
Trying caller script 'tb9600-ec' from '/etc/uucp/Dialers'
expect: (“”)
```

The host waits a couple of seconds and then sets the registers on the modem. The information shown in the log below is modem-specific. It comes from the `/etc/uucp/Dialers` file.

```
got it
sendthem (DELAY)
APAUSE
APAUSE
APAUSE
T&D2E1V1X1Q0S2=255S12=255S50=6S58=2^M<NO CR>)
```

The next lines are the dialog between the modem and the host machine. `expect (OK^M)` means that the host expects the modem to send an ”okay.” The words `got it` at the end of the second line indicate that the host got the “okay” message from the modem.

```
expect: (OK^M)
AAAT&D2E1V1X1Q0S2=255S12=255S50=6S58=2^M^M^JOK^Mgot it
```

Next, the host sends the string below to modem, which does the actual dialing. The phone number in the second line is retrieved from the entry for the remote host in the `/etc/uucp/Systems` file..

```
sendthem (ECHO CHECK ON
A^JATTDDTT99003300887744^M^M<NO CR>)
```

The line beginning with `expect` indicates that the local host expects to get a response from the modem at a speed of 9600 bps. The next line indicates that the modem responded.

```
expect: (CONNECT 9600)
^M^JCONNECT 9600got it
```

This line indicates that hardware flow control has started on the link. The host obtains flow control information from the `/etc/uucp/Dialers` file.

```
STTY crtscts
```

In the next series of messages, the local host waits for the remote host to send it a standard UNIX login prompt.

```
getty ret 8
expect: (“”)
got it
sandiest (^J^M)
expect: (login:)
```

The next messages indicate that the local host has received the login prompt from the remote. It then retrieves the appropriate login sequence from the chat script in the `/etc/uucp/Systems` entry for the remote host. This sequence is `Ppong^M`, which is required for login by the remote host.

```
^M^J^M^Jlogin:got it
sendthem (Ppong^M)
```

In these messages, the local host waits for the `ssword` prompt from the remote host. Upon receipt of the prompt, the local host sends the password retrieved from the chat script in the `/etc/uucp/Systems` entry for the remote host.

```
expect: (ssword:)
login: Ppong^M^JPassword:got it
```

The following messages indicate that dialing and modem connection completed successfully.

```
sendthem (ppptest1^M)
call cleanup(0)^M
```

## *Communications Between the Local and Remote Hosts*

At this point, PPP communications start, as the link between local and remote hosts is now established.

The first lines in this part of the session constitute a *configuration request* (`Config-Req`). This is the first PPP packet sent to the remote host. The configuration request is one example of a Link Control Protocol (LCP) packet. It requests that configuration be set up and then sets up the PPP link between endpoint machines.

*Code Example 10-4* Configuration Request

```
11:54:20 004298 ipdptp0 SEND PPP ASYNC 29 Octets LCP Config-Req
ID=4c LEN=24 MRU=1500 ACCM=00000000 MAG#=69f4f5b2 ProtFCOMP
AddrCCOMP
```

Here is a description of the configuration request shown in Code Example 10-4.

- `11:54:20` – Time stamp field, indicating the time when the packet was sent
- `004298` – Number of the packet
- `ipdptp0` – Network interface used
- `SEND PPP ASYNC` – Indicates that the modem is sending asynchronous PPP
- `29 Octets` – Amount of data the host sent
- `LCP` – Packet type to send
- `ID=4c` – Identifier associated with the packet; it is actually part of the packet
- `LEN=24` – Length of the LCP part of the packet

The remaining items are a list of options to be negotiated between hosts.

- `MRU=1500` – Maximum Receive Unit (MRU), the largest packet size the calling host can receive from the remote host
- `ACCM=00000000` – Asynchronous Character Map (ACCM), the mask sent to the remote host that tells what control characters to escape on transmission

- `MAG#=69f4f5b2` – Magic number field; used for loopback detection mechanism
- `ProtFCOMP AddrCCOMP` – Asks for the remote host to compress certain parts of the frame header (protocol field, address field).

The next lines are reporting invalid PPP packets. They come from the remote host, which actually is sending out UNIX text. This does not indicate a problem with PPP.

```
11:54:20 004299 ipdptp0 RECEIVE {Invalid ppp packet}PPP ASYNC 7
Octets [BAD FCS] {Unrecognized protocol:    1}

11:54:20 004299 ipdptp0 RECEIVE PPP ASYNC 73 Octets [BAD FCS]
{Unrecognized protocol:    880a}
```

In these packets, the local host receives the remote host's request for configuration, and then sends out another configuration request. The packets are identical except for their ID fields. The ID field helps to distinguish between the two packets.

```
11:54:21 004301 ipdptp0 RECEIVE PPP ASYNC 29 Octets LCP Config-
Req  ID=35 LEN=24 MRU=1500 ACCM=00000000 MAG#=a8562e5f ProtFCOMP
AddrCCOMP
11:54:21 004302 ipdptp0 SEND PPP ASYNC 29 Octets LCP Config-Req
ID=4d LEN=24 MRU=1500 ACCM=00000000 MAG#=69f4f5b2 ProtFCOMP
AddrCCOMP
```

In this packet, the local host acknowledges the remote request by sending it a configuration acknowledgment (`Config-ACK`).

```
11:54:21 004303 ipdptp0 SEND PPP ASYNC 29 Octets LCP Config-ACK
ID=35 LEN=24 MRU=1500 ACCM=00000000 MAG#=a8562e5f ProtFCOMP
AddrCCOMP
```

The local host receives a configuration request (`Config-Req`) from the remote host.

```
11:54:21 004304 ipdptp0 RECEIVE PPP ASYNC 29 Octets LCP Config-
Req  ID=36 LEN=24 MRU=1500 ACCM=00000000 MAG#=a8562e5f ProtFCOMP
AddrCCOMP
```

In these packets, the local host acknowledges the second packet sent by the remote host and receive the remote host's acknowledgment.

```
11:54:21 004305 ipdptp0 SEND PPP ASYNC 29 Octets LCP Config-ACK
ID=36 LEN=24 MRU=1500 ACCM=00000000 MAG#=a8562e5f ProtFCOMP
AddrCCOMP
11:54:21 004306 ipdptp0 RECEIVE PPP ASYNC 29 Octets LCP Config-
ACK  ID=4d LEN=24 MRU=1500 ACCM=00000000 MAG#=69f4f5b2 ProtFCOMP
AddrCCOMP
```

Here the local host negotiates parameters about IP transmission. `LEN=16` gives the packet size. `VJCOMP` indicates Van Jacobsen header compression. `IPADDR` is followed by the calling host's IP address.

```
11:54:21 004307 ipdptp0 SEND PPP ASYNC 21 Octets IP_NCP Config-
Req  ID=4e LEN=16 VJCOMP MAXSID=15 Sid-comp-OK IPADDR=192.9.68.70
```

This packet indicates that the local host has received IP configuration from the remote host, including its IP address.

```
11:54:22 004308 ipdptp0 RECEIVE PPP ASYNC 21 Octets IP_NCP
Config-Req  ID=37 LEN=16 VJCOMP MAXSID=15 Sid-comp-OK
IPADDR=192.9.68.71
```

The local host sends this ACK to the remote host and receives an ACK from the remote host.

```
11:54:22 004309 ipdptp0 SEND PPP ASYNC 21 Octets IP_NCP Config-
ACK  ID=37 LEN=16 VJCOMP MAXSID=15 Sid-comp-OK IPADDR=192.9.68.71
11:54:22 004310 ipdptp0 RECEIVE PPP ASYNC 21 Octets IP_NCP
Config-ACK  ID=4e LEN=16 VJCOMP MAXSID=15 Sid-comp-OK
IPADDR=192.9.68.70
```

The first message below indicates that IP has started on the link. The next message indicates that the local host is sending IP traffic over the link.

```
11:54:22 start_ip: IP up on interface ipdptp0, timeout set for
120 seconds
11:54:24 004311 ipdptp0 SEND PPP ASYNC 89 Octets IP_PROTO
```

In the first message below the local host receives IP traffic from the remote host. The subsequent messages indicate that the interface was disconnected because of an idle time-out.

```
11:54:25 004312 ipdptp0 RECEIVE PPP ASYNC 89 Octets IP_PROTO
11:56:25 process_ipd_msg: interface ipdptp0 has disconnected
11:56:25 disconnect: disconnected connection from  ipdptp0
```

The next messages begin the termination sequence. The first message indicates that the remote host has sent a packet to terminate the IP layer. The second is the local host's acknowledgment of the request to terminate.

```
11:56:25 004313 ipdptp0 RECEIVE PPP ASYNC 9 Octets IP_NCP Term-
REQ  ID=38 LEN=4
11:56:25 004314 ipdptp0 SEND PPP ASYNC 9 Octets IP_NCP Term-ACK
ID=38 LEN=4
```

The local host receives a request to terminate the LCP layer. The second message is an acknowledgment of the request, causing a graceful shutdown.

```
11:56:25 004315 ipdptp0 RECEIVE PPP ASYNC 9 Octets LCP Term-REQ
ID=39 LEN=4
11:56:25 004316 ipdptp0 SEND PPP ASYNC 9 Octets LCP Term-ACK
ID=39 LEN=4
```

This message indicates that the link has closed.

```
11:56:29 004317 ipdptp0 PPP DIAG CLOSE
```

# Tailoring Your PPP Link 11

This chapter contains information you will need to configure PPP links less commonly used than the basic links described in Chapter 9, “Configuring PPP.” The text includes instructions for configuring two types of PPP links: the dial-in server with dynamic point-to-point links and the virtual network, which uses multipoint links. The chapter concludes with tables listing all available keywords for the `asppp.cf` configuration file.


| | |
|---|---|
| *Addressing Issues for Dynamically Allocated Links* | *page 167* |
| *Editing asppp.cf for Dynamic Link* | *page 169* |
| *Addressing Issues for Virtual Networks* | *page 173* |
| *asppp.cf Configuration File for a Virtual Network* | *page 174* |
| *Editing asppp.cf for PAP/CHAP Security* | *page 176* |
| *Configuration Keywords* | *page 180* |


## Configuring Dynamically Allocated PPP Links

A dial-in server with a dynamic point-to-point link gives your site all the advantages of point-to-point communications. Chapter 7, “Understanding PPP,” introduces this configuration type. It consists of remote hosts communicating with at least one dial-in server that dynamically allocates point-to-point links on an as-needed basis. The sample configuration shown in Figure 11-1 on page 166 will be used throughout this section.

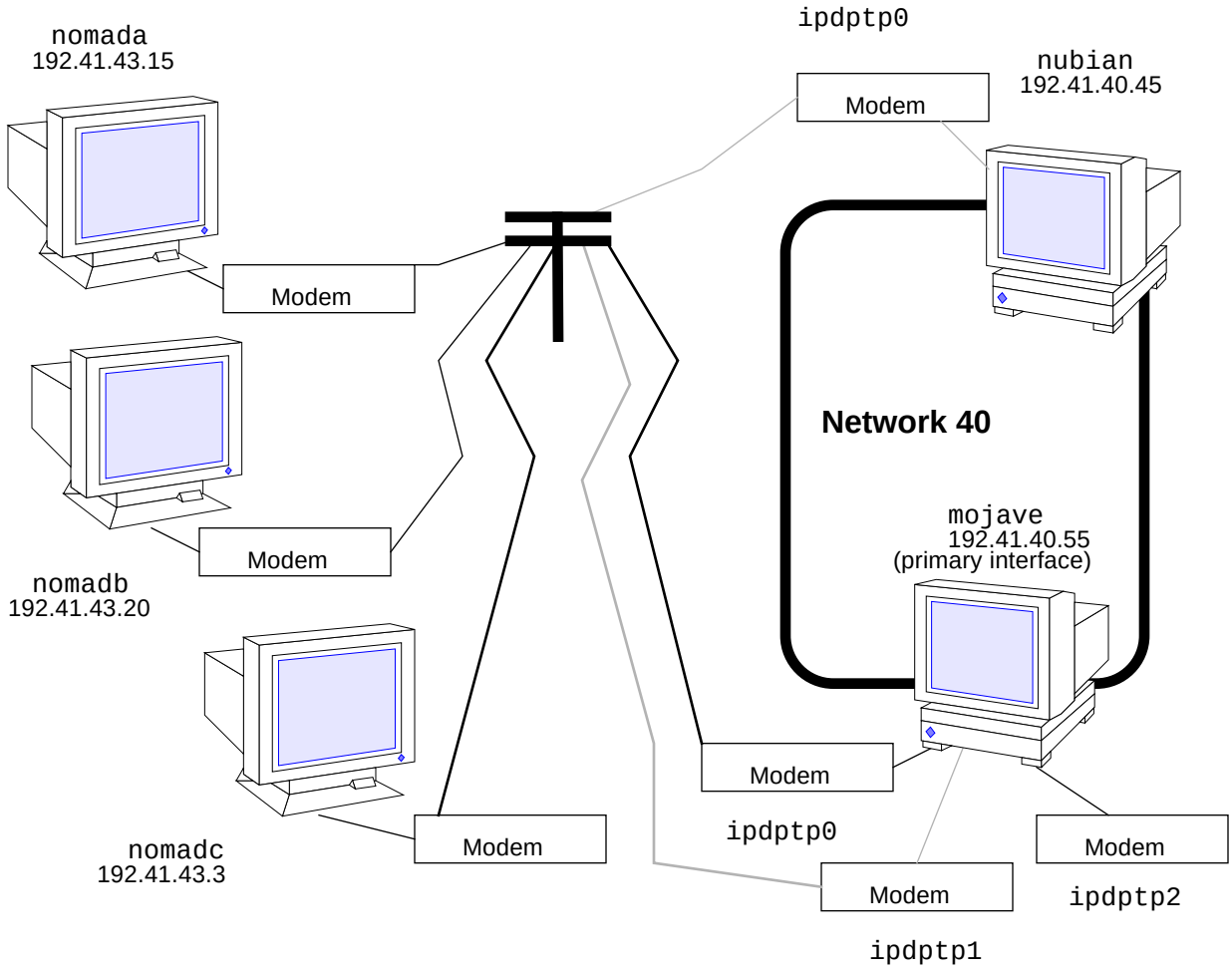



*Figure 11-1* Network of Remote Hosts and Dynamic Link Dial-in Servers

Each remote host communicates with the dial-in server using a standard point-to-point link. However, unlike the multipoint dial-in server in Figure 9-1 on page 134, dial-in servers `mojave` and `nubian` connect to a calling host over a dynamic point-to-point link. One of the servers allocates an available link whenever a remote host attempts to establish a connection.

You use the same generic procedures for configuring dynamic links as you do for the remote host–to–multipoint dial-in server link, as described in ”Overview of the Configuration Process” on page 131. However, the dynamic point-to-point link has its own set of issues and requires slightly different modifications to the files involved in configuration.

## *Addressing Issues for Dynamically Allocated Links*

You must add host information to the `/etc/inet/hosts` file for each machine that will use the dynamically allocated PPP link. The IP addresses for the PPP endpoints should follow these conventions:

- For the dial-in servers, you must use the IP address of the server’s primary network interface (for example `le0` or `smc0`) as the address of the dynamic link.
- For a remote host, you can use the IP address of the host’s primary network interface as the address of its point-to-point link, or you can designate a PPP-specific IP address for its point-to-point link.

## *Updating the* `hosts` *Database for Dynamic Links*

You must update the `hosts` database on all machines involved in the dynamic-link configuration.

### ▼ How to Update a Remote Host

When configuring the `hosts` databases on the remote machines, do the following:

1. **Add to the `/etc/inet/hosts` file the IP address and host name of the primary network interface for each dial-in server on the other end of the link.**
   For example, in Figure 11-1 on page 166, `nomada` will have in its `/etc/inet/hosts` file the IP addresses of the primary network interfaces of dial-in servers `nubian` and `mojave`. The `/etc/inet/hosts` files for `nomadb` and `nomadc` would contain the same information about the dial-in servers.

The /etc/inet/hosts file on nomadc might look like:

```
# Internet host table
#
127.0.0.1          localhost       loghost
192.41.43.3        nomadc
192.41.40.45       nubian
192.41.40.55       mojave
```

2. **Add to the /etc/inet/hosts file the IP addresses of any machines on the dial-in server's physical network that the remote host can remotely log in to.**
3. **Update the databases on any name server on the physical network with the host names and IP addresses of the remote hosts.**

### ▼ How to Update the Dial-in Server

You do not have to add any PPP-specific address to the hosts database for the dial-in server. The dynamically allocated link must use the server's primary network interface. Therefore, when configuring the hosts database for the dial-in server, do the following:

1. **Add entries to the server's /etc/inet/hosts files for each remote host served.**
2. **Add to the /etc/inet/hosts files of every machine on the physical network entries for any remote hosts they are permitted to communicate with.**

## *Considerations for Other Files*

The next steps in the configuration process involve editing the /etc/passwd file and the /etc/shadow file. Edit these files for the dynamic–link configurations just as you would for the remote host–to–multipoint dial-in server configuration. Refer to "Modifying the /etc/passwd File" on page 138 for information regarding the /etc/passwd and /etc/shadow files.

## *Editing* `asppp.cf` *for Dynamic Link*

The `asppp.cf` configuration file for a dynamic–link configuration must contain information about remote hosts and the interfaces to use for the PPP link. After the dial-in server boots, its link manager uses this information to establish communications whenever the server is called by a remote endpoint.

**Note –** Each remote host sets up the point-to-point link described in ”Parts of Basic Configuration File” on page 140.

### *Dial-in Server With Dynamic Link*

When the dial-in server receives an incoming packet, the link manager reads the `path` sections of its configuration file to identify the remote endpoint and determine the interface to use. The configuration file shown in Code Example 11-1 does not contain an interface keyword. Instead, the link manager uses interface information established in the `defaults` section.

The `asppp.cf` configuration file for a dial-in server with dynamically allocated links might resemble Code Example 11-1 on page 170:

*Code Example 11-1* Configuration File for Server With Dynamically Allocated Link

```
ifconfig ipdptp0 plumb mojave down
ifconfig ipdptp1 plumb mojave down
ifconfig ipdptp2 plumb mojave down

# This means grab whatever interface is available (not in use)
defaults
    interface ipdptp*

# Each path specifies a machine that might dial up / log
# in to this server

path
    peer_system_name tamerlane   # nomada uses the login name
                                 # tamerlane
    peer_ip_address nomada

path
    peer_system_name lawrence   # nomadb uses the name lawrence
                                # for login
    peer_ip_address nomadb

path
    peer_system_name nomadc
    peer_ip_address azziz
```

### `ifconfig` *Section for Server With Dynamic Links*

The `ifconfig` section for a dial-in server with a dynamically allocated link has the syntax:

`ifconfig ipdptp`*n* `plumb` *server-name* `down`

Code Example 11-1 contains three `ifconfig` lines, each initializing a point-to-point interface.

```
ifconfig ipdptp0 plumb mojave down
ifconfig ipdptp1 plumb mojave down
ifconfig ipdptp2 plumb mojave down
```

Each interface has a corresponding modem attached to the dial-in server `mojave`, as shown in Figure 11-1 on page 166. This enables `mojave` to have up to three point-to-point links active at one time. Note that since PPP interfaces are virtual (not associated with any hardware), you could just as easily configure three `ipdptp` interfaces on a server with only one modem, as shown on `nubian`.

### `defaults` *Section for Server With Dynamic Links*

When you configure a dynamically allocated link, you may want to include a `defaults` section in the `asppp.cf` file. This section sets the defaults for the value replacing *keyword*, wherever *keyword* subsequently appears in the `asppp.cf` file. The syntax for the `defaults` section is:

```
default
        keyword
```

Code Example 11-1 on page 170 uses the keyword `interface` to define the interface as `ipdptp*`, indicating a dynamic link. The asterisk wildcard tells the link manager to use any available `ipdptp` interface defined in the `ifconfig` section. Thus the link manager on server `mojave` will use either `ipdptp0`, `ipdptp1`, or `ipdptp2`—whichever is the first interface configured “down” that it finds.

### `path` *Section for Server With Dynamic Links*

The configuration file for the server with dynamic links must contain `path` sections for every remote host permitted to establish connections with the server. The path section has the following syntax:

```
path
        peer_system_name endpoint-username
        peer_ip_address endpoint-hostname
```

Note that there is no `interface` keyword defined in the `path` section because this value is defined in the defaults section.

The `peer_system_name` and `peer_ip_address` keywords have the same meaning here as they do in the configuration file for the multipoint server. See ”path Section for Multipoint Dial-in Server” on page 144 for more information.

### *Additional Keywords*

You can supply other keywords in the `asppp.cf` file to define how endpoint machines should communicate, including the use of security keywords as explained in ”Configuration Keywords” on page 180.

## *Configuring a Virtual Network*

Virtual networks consist of a group of standalone computers, each in an isolated location, that can connect to each other through PPP multipoint links. ”Virtual Networks” on page 110 introduces virtual network concepts. This section explains how to configure a virtual network.

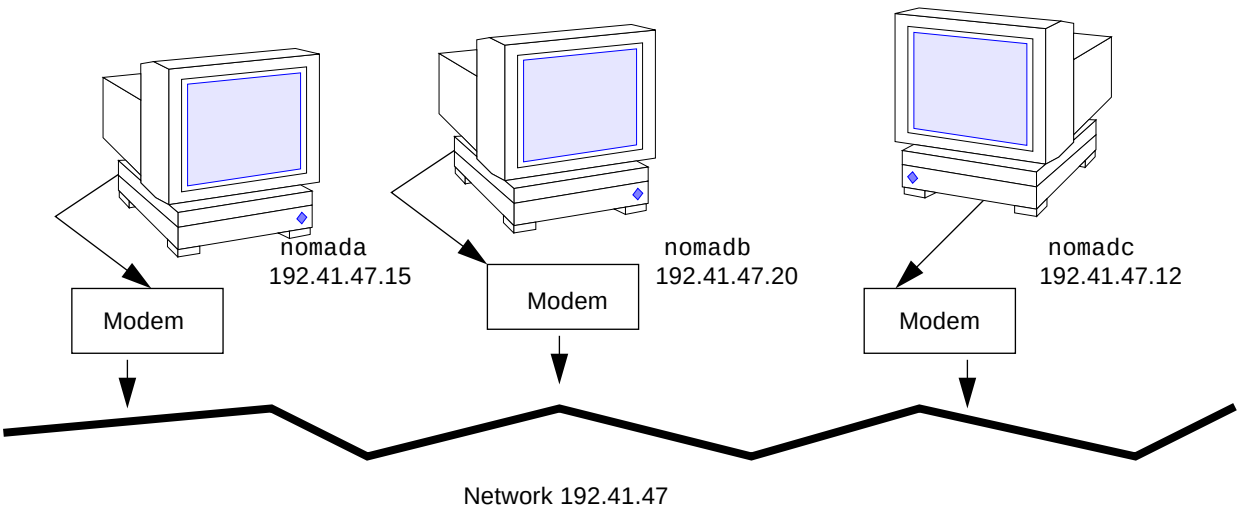



*Figure 11-2* Sample Virtual Network

The network shown in Figure 11-2 consists of three isolated computers. Each member of the network connects to the other members of the network through a multipoint PPP link. Therefore, to create such a network, you (and perhaps other network administrators at the remote location) have to configure a multipoint PPP link on each participating host.

You use the same generic process for configuring multipoint links as you do for configuring a multipoint dial-in server link, as described in ”Overview of the Configuration Process” on page 131. However, the virtual network has its own set of issues and requires you to configure each host in the network accordingly.

## *Addressing Issues for Virtual Networks*

You must add host information to the `/etc/hosts` file for each machine in the virtual network. When typing the IP addresses used for the PPP endpoints,

- Designate a PPP-specific IP address for its point-to-point link. Note that if the machine was not previously configured in a physical network, you will have to create an IP address for the PPP link. This address becomes the host’s primary network interface.
- Create a network number for the virtual network. See ”Assigning a Network Number to the PPP Link” on page 126 for more information.

## *Updating* `hosts` *and* `networks` *Databases*

The first step in the configuration process involves updating the `hosts` and `networks` databases with information about your virtual network.

### `/etc/inet/hosts` *File for the Virtual Network*

The `/etc/inet/hosts` file on each machine must contain the addressing information for every member of the network that this host has permission to access. For example, each host in the network in Figure 11-2 on page 172 would have this information:

```
# Internet host table
#
127.0.0.1          localhost      loghost
192.41.47.15        nomada
192.41.47.20        nomadb
192.41.47.12        nomadc
```

### `/etc/inet/networks` *File for the Virtual Network*

Since the virtual network requires a unique IP address, you will have to type this address in the `networks` database. For example, the network shown in Figure 11-2 has the number 192.41.47. Moreover, if the hosts on the network need to communicate with other networks, you should register the network with the InterNIC addressing authority. See Chapter 4, “Configuring TCP/IP on the Network,” for information on editing the `networks` database.

Each host on the virtual network must have an entry with the network’s address in the `/etc/inet/networks` file. For example, each host on network 192.41.47 might have the following in `/etc/inet/networks`:

```
# Internet networks
#
# arpanet   10          arpa
# ucb-ether 46          ucbether
#
# local networks
loopback    127
ppp         192.41.47   #remote sales offices
```

## *Considerations for Other Files*

The next steps in the configuration process involve editing the UUCP databases, the `/etc/passwd` file, and the `/etc/shadow` file. You edit these files for the machines in the virtual network just as you would for the multipoint dial-in server configuration. Refer to ”Editing UUCP Databases” on page 136 for UUCP-related information and ”Modifying the /etc/passwd File” on page 138 for information regarding the `passwd` file.

## `asppp.cf` *Configuration File for a Virtual Network*

The configuration file for a local machine on a virtual network must contain information about all remote hosts on the network that the local host can access. Moreover, each machine on the virtual network must be configured for both dial-in and dial-out functions. After the local machine boots, its link manager reads the `asppp.cf` file to establish communications.

Code Example 11-2 shows a configuration file such as you would set up for nomada on a virtual network 192.41.47.

*Code Example 11-2* Configuration File for nomada

```
# /etc/asppp.cf for hosta

ifconfig ipd0 plumb nomada netmask + up
defaults
    interface ipd0
path
    peer_ip_address  nomadb
    peer_system_name lawrence   # name machine logs in with
path
    peer_ip_address nomadc
    peer_system_name azziz
```

Code Example 11-3 shows a configuration file such as you would set up for nomadb on virtual network 192.41.47.

*Code Example 11-3* Configuration File for nomadb

```
# /etc/asppp.cf for nomadb

ifconfig ipd0 plumb nomadb netmask + up
defaults
    interface ipd0
path
    peer_ip_address   nomada
    peer_system_name  tamerlane  # name the machine logs in with
path
    peer_ip_address   nomadc
    peer_system_name  azziz
```

# 11

## *Editing* asppp.cf *for PAP/CHAP Security*

You can edit the asppp.cf file to establish security and to specify whether parts of the link will respond to Password Authentication Protocol (PAP), or Challenge-Handshake Authentication Protocol (CHAP) as described in "PPP Security" on page 116. The asppp.cf file is edited by adding a series of keywords. In this section *authenticator* is the system starting the link or challenge, and is frequently the server. *Peer* is the other end of the link, and is often the client.

The keywords to be added are require_authentication and will_do_authentication. The authenticator or server will generally require authentication and the peer or client will generally do authentication.

*Table 11-1* Authenticator Keywords and Associated Strings

| require_authentication pap | require_authentication chap |
|---|---|
| pap_peer_id | chap_peer_secret |
| pap_peer_password | chap_peer_name |

*Table 11-2* Peer Keywords and Associated Strings

| will_do__authentication pap | will_do_authentication chap |
|---|---|
| pap_id | chap_secret |
| pap_password | chap_name |

### ▼ How to Install PAP/CHAP

1. **On the server, become superuser and prepare to edit the** /etc/asppp.cf **file.**
2. **Add the** require_authentication **keyword for each machine on the link to use either CHAP or PAP security.**
   a. **For each pap keyword add an associated** pap_peer_id **and** pap_peer_password **string.**

**b. For each chap keyword add an associated** `chap_peer_secret` **and** `chap_peer_name` **string.**

You can state the keywords explicitly, or if you prefer, you can use the default for the path. Refer to Table 11-3 on page 178 to see what each keyword specifies. Examples can be found on page 179.

3. **On each remote host on the link to use either PAP or CHAP security, add an entry in the remote host's** `/etc/asppp.cf` **file with the** `will_do_authentication` **keyword.**

   **a. For each pap keyword entry add an associated** `pap_id` **and** `pap_password` **string.**

   **b. For each chap keyword entry add an associated** `chap_secret` **and** `chap_name` **string.**

   You can state the keywords explicitly, or if you prefer, you can use the default for the path. Refer to Table 11-3 on page 178 to see what each keyword specifies. Examples can be found on starting on page 179.

### *Rules for PAP/CHAP Keywords*

- Either server or client can require authentication or offer to do authentication.
- If PAP and CHAP are both present, the authenticator will first try CHAP. If that fails it will try PAP. If both fail, the link is terminated.
- The default value for PAP and CHAP authentication keywords is off. The syntax for keywords is:

  `require_authentication  off|pap[chap]|chap[pap]`

  `will_do_authentication  off|pap[chap]|chap[pap]`

- If you fail to specify `pap_id` and `pap_password` or `pap_peer_id` and `pap_peer_password` keywords and values for the associated path, the corresponding values are set to the NULL string.
- You must specify `chap_name`,`chap_secret`,`chap_peer_secret` and `chap_peer_name` keywords and values for that path.

*Table 11-3* PAP / CHAP Keyword Definitions

| Keywords | Value Definition |
|---|---|
| `require_authentication` *keywords* [1] | Specifies whether the peer must authenticate itself. If either `pap` or `chap` is present, the peer must participate in authentication or end the connection. The default value is off. |
| `pap_peer_id` *peername*[2] | Specifies the name of the peer to be authenticated for the current path. *peername* string is one or more octets[4]. To indicate a zero-length string, do not include the keyword. |
| `pap_peer_password` *string*[3] | Specifies password for peer in one or more octets. To indicate a zero-length string, do not include the keyword. |
| `chap_peer_secret` *string* | Specifies the secret used with the challenge value to generate the response sent by the peer. The format is one or more octets, preferably at least 16. |
| `chap_peer_name` *peername* | Specifies the identity of the peer transmitting the packet. The name should not be NULL or terminated with CR/LF. The name is received from the peer in a response packet and consists of one or more octets. |
| `will_do_authentication` *keywords* | Specifies whether the system is willing to participate as the authenticated peer in the specified authentication process. If both pap and chap are present, then the system is willing to participate in either authentication protocol. The default value is off. |
| `pap_id` *peername* | Specifies the name of the system to be sent to the authenticator in the response packet. To indicate a zero-length string, do not include the keyword. |
| `pap_password` *string* | Specifies the password for the system to be sent to the authenticator in the response packet. To indicate a zero-length string, do not include the keyword. |
| `chap_secret` *string* | Contains the secret that is used with the received challenge value to generate the response sent to the authenticator. The format is one or more octets, preferably at least 16 |
| `chap_name` *peername* | Specifies the identity of the system. The name should not end with a NULL or CR/LF. The name is sent to the authenticator in a response packet. |

1. Keyword alternatives are: off | pap[chap] | chap[pap]

2. *peername* is the name of the system at the other end of the point-to-point link from the authenticator. It takes the form of a string with the syntax specified in footnote 3.

3. *string* is a single token without embedded white space. The standard ANSI C \ escape sequence may be used to embed special characters. Use \s for the space character. Any pound sign at the beginning of the string must be escaped (\#) to avoid interpretation as a comment. A NULL (\0) will truncate the string.

4. Octet is the more accurate definition of byte.

## *PAP/CHAP Examples*

This code example shows the `asppp.cf` file for the server `mojave` with PAP and CHAP authentication required. The peers are `tamerlane` (PAP) and `lawrence` (CHAP).

*Code Example 11-4* Code Example for Server `mojave`

```
ifconfig ipdptp0 plumb mojave down
ifconfig ipdptp1 plumb mojave down
defaults
      interface ipdptp*
path
      peer_system_name tamerlane
      require_authentication pap  #tells tamerlane that mojave
                                  #requires pap authentication
      pap_peer_id desert
      pap_peer_password oasis
path
      peer_system_name lawrence
      require_authentication chap  #tells lawrence that mojave
                                   #requires chap authentication
      chap_peer_name another\sdesert
      chap_peer_secret secret\soasis\swith\007bell
```

The next code sample shows `mojave`'s remote host `tamerlane` offering to do both PAP and CHAP authentication.

*Code Example 11-5* Code Example for Remote Host `tamerlane`

```
ifconfig ipdptp0 plumb tamerlane mojave up
path
      interface ipdptp0
      peer_system_name mojave
      will_do_authentication chap pap #tamerlane tells mojave
                                      #that it will do chap and
                                      #pap authentication
      pap_id desert
      pap_password oasis
      chap_name desert\srain
      chap_secret %$#@7&*(+|‘P’12
```

*Code Example 11-6* Code Example for Remote Host `lawrence`

```
ifconfig ipdptp0 plumb lawrence mojave private up
path
     interface ipdptp0
     peer_system_name mojave
     will_do_authentication chap  #lawrence tells mojave that it
                                  #will do chap authentication
     chap_name another\sdesert
     chap_secret secret\soasis\swith\007bell
```

The code sample above shows `mojave`’s remote host `lawrence` offering to do CHAP authentication.

Ideally, both CHAP and PAP are included in the configuration file, with the server requiring authentication and the remote host willing to do authentication. However this is reversible so that either side can require authentication. CHAP secrets need to be delivered by secure means. This generally involves handing them over in person.

## *Configuration Keywords*

This section describes the configuration keywords available for the `asppp.cf` configuration file and the values you must define for them. Most of these keywords are optional. The required ones are indicated. For further explanations of the keywords, refer to RFCs 1331, 1332, 1333, and 1334.

Table 11-4 lists required keywords that must appear in all `asppp.cf` configuration files.

*Table 11-4* Required Keywords for `asppp.cf`

| **Keywords** | **Value Definitions** |
|---|---|
| `ifconfig` *parameters* | Tells the link manager to run the `ifconfig` command with the values supplied by *parameters*. See "ifconfig Section of the asppp.cf File" on page 140, "ifconfig Section for Multipoint Dial-in Server" on page 143, and the `ifconfig` (1M) man page for more information. |
| `path` | Specifies the beginning of the token sequences that are grouped together as attributes of this (current) path. The collection of attributes comprising the current path are terminated by the occurrence of a subsequent `path` keyword, `defaults` keyword, or by the end–of–file charachter. |
| `interface` (`ipdptp`*n*, `ipdptp*`, or `ipd`*n*) | Specifies either an `ipdptp` (static point-to-point), `ipdptp*` (dynamic point-to-point), or `ipd` (multipoint) device for each interface in your network. For `ipdptp`*n* and `ipd`*n*, this keyword associates the specific interface defined by *n* with the current path. *n* must be a nonnegative integer. It matches the interface defined in the `path` section with the interface stated in the `ifconfig` section.<br><br>For the `ipdptp*` interface, the * indicates that the interface will match any point-to-point interface that is configured as "down." |
| `peer_system_name` *hostname*<br>`peer_system_name` *username* | On dial-out machines, specifies the *hostname* of the remote endpoint that the local machine wants to call. This is the same as the system name in the `/etc/uucp/Systems` file. Associates the remote system name with the current path. This name is used to look up modem- and peer-specific information for outbound connections in the `/etc/uucp/Systems` file.<br><br>On dial-in machines, this specifies the *username* that remote machines will use when logging in to the dial-in machine. The appropriate path is determined by matching *username* with the login name that was used to obtain the connection. |
| `peer_ip_address` *hostname*<br>`peer_ip_address` *ip-address* | Specifies the destination host address. It is required only for multipoint links. This address is associated with the current path. The value is ignored if the path specifies a point-to-point interface. The address format may be dotted decimal, hexadecimal, or symbolic. |

Table 11-5 contains optional keywords for `asppp.cf` that you can use to further define your PPP configuration.

*Table 11-5* Optional Keywords for `asppp.cf`

| Keywords | Value Definitions |
|---|---|
| `debug_level` *0–9* | The integer between 0–9 defines how much debugging information should be written to the log file. The higher the number, the more output is generated. |
| `defaults` | Indicates that all following token sequences up to the next `path` keyword, or the end-of-file character, set default attributes that affect subsequently defined paths. |
| `default_route` | Tells the link manager to add the path's peer IP address to the route table as the default destination when the IP layer corresponding to the current path is fully operational. The route is removed when the IP layer is brought down. |
| `inactivity_timeout` *seconds* | Specifies the maximum number of seconds that the connection for the current path can remain idle before it is terminated. A zero may be specified to indicate no timeout. The default is 120 seconds. |
| `ipcp_async_map` *hex-number* | Specifies the asynchronous control-character map for the current path. *hex-number* indicates the natural (big-endian) form of the four octets that comprise the map. The default value is 0x FFFFFFFF. |
| `ipcp_compression` (`vj` or `off`) | Specifies whether IP compression is enabled. The Van Jacobson compression algorithm (`vj`) is the default. |
| `lcp_compression` (`on` or `off`) | Specifies whether PPP address, control, and protocol field compression is enabled. The default is `on`. |
| `lcp_mru` *number* | Specifies the value of the desired maximum receive unit packet size. The number is the size in octets. The default is 1500. |
| `negotiate_address` (`on` or `off`) | Indicates whether local IP address assignment is obtained through negotiation and assigned dynamically or not. If enabled, the local address will be obtained from the remote end of the PPP link. If so obtained, any local address other than 0.0.0.0 can be used to initially configure the interface. The default is not to negotiate (off). |
| `peer_ip_address` *hostname*<br>`peer_ip_address` *ip-address* | Specifies the destination host address. This keyword is optional for point-to-point links only. *address* is associated with the current path. The address format may be dotted decimal, hexadecimal, or symbolic. |
| `version` *n* | Specifies that the contents of the configuration file correspond to format version *n*. If this keyword is present, it must be the first keyword in the file. If absent, the version is assumed to be 1. This book contains the definition of the version 1 format for the configuration file. |

# *Part 3 — Administering UUCP Communications*

The UUCP file transfer system enables you to send files and electronic mail from one UNIX-based system to another. This part explains how to administer the complex UUCP system.

The materials in this part assume that you are a very experienced network administrator with some practical knowledge of modem administration and wide-area networks. The text assumes that if you are going to use UUCP over a telephone line, you are familiar with the procedures for adding hardware to your computer, have already connected modems to your machines, and are able to use `tip` or `cu` to dial out.


| | |
|---|---|
| *Software Comprising UUCP* | *page 186* |
| *Introducing the UUCP Database Files* | *page 189* |
| *Running UUCP Over TCP/IP* | *page 236* |
| *UUCP Error Messages* | *page 241* |

# UUCP Databases and Programs 12

This chapter introduces the UUCP programs and daemons. It then provides complete information for setting up the UUCP database files as part of UUCP configuration. Chapter 13, “Configuring and Maintaining UUCP,” explains how to configure UUCP after the databases have been created.


| | |
|---|---|
| *UUCP Hardware Configurations* | *page 186* |
| *Software Comprising UUCP* | *page 186* |
| *Introducing the UUCP Database Files* | *page 189* |
| */etc/uucp/Systems File* | *page 191* |
| */etc/uucp/Devices File* | *page 198* |
| */etc/uucp/Dialers File* | *page 206* |
| *Other Basic Configuration Files* | *page 211* |
| */etc/uucp/Permissions File* | *page 214* |


The UNIX-to-UNIX Copy Program (UUCP) enables computers to transfer files and exchange mail with each other. It also enables computers to participate in large networks such as Usenet.

The Solaris environment 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 comprise the system, of which the program `uucp` is only a part. The UUCP utilities range from those used to copy files between computers (`uucp` and `uuto`) to those used for remote login and command execution (`cu` and `uux`)

# 12

## 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 where two computers communicate regularly and are physically close—within 50 feet of each other. You can use a limited distance–modem to increase this distance somewhat.
- *Telephone lines* – 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 requested by UUCP. The recipient machine must have a modem capable of answering incoming calls.
- *Network* – UUCP can also communicate over a network running TCP/IP or other protocol family. Once your computer is established as a host on a network, it can contact any other host 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 *System Administration Guide, Volume II* and the manuals that came with the modem for assistance.

## Software Comprising UUCP

The UUCP software is automatically included when you run the Solaris installation program and select the entire distribution. Alternatively, you can add it using `pkgadd`. The UUCP programs can be divided into three categories: daemons, administrative programs, and user programs.

### 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 used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers data and execute files, logs results, 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.

`uucp`, `uuto`, and `uux` programs execute the `uucico` daemon after all the required files have been created, to contact the remote computer. `uusched` and `Uutry` all execute `uucico`. (See the `uucico`(1M) man page for details.)

- `uuxqt` – Executes remote execution requests. It 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. It 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 will be 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` will then prompt for a password, unless one is not required. (See the `in.uucpd`(1M) man page for details.)

## *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. It 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. It 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. It invokes the `uucico` daemon to establish a communication link between your machine and the remote computer you specify. (See the `Uutry`(1M) man page for details.)
- `uucheck` – Checks for the presence of UUCP directories, programs, and support files. It can also check certain parts of the `/etc/uucp/Permissions` file for obvious syntactic errors. (See the `uucheck`(1M) man page for details.)

## *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 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. It 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 it 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 using `uuto`. (See the `uuto`(1C) man page.)
- `uux` – Creates the work, data, and execute files 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`). It also provides a means of controlling queued transfers. (See the `uustat`(1C) man page for details.)

## *Introducing the UUCP Database Files*

A major part of UUCP setup is the configuration of the files comprising the UUCP database. These files are in the `/etc/uucp` directory. You need to edit them to set up UUCP or PPP on your machine. The files include:

- `Config` – Contains a list of variable parameters. You can manually set these parameters to configure the network.
- `Devconfig` – Used to configure network communications.
- `Devices` – Contains information concerning the location and line speed of automatic call unit (modem), direct links, and network devices. It is used by PPP as well as UUCP.
- `Dialers` – Contains character strings required to negotiate with modems to establish connections with remote computers. It is used by PPP as well as UUCP.
- `Dialcodes` – Contains dial-code abbreviations that may be used in the phone number field of `Systems` file entries. Though not required, it can be used by PPP as well as UUCP.
- `Grades` – Defines job grades, and the permissions associated with each job grade, that users may specify to queue jobs to a remote computer.
- `Limits` – Defines the maximum number of simultaneous `uucico`s, `uuxqt`s, and `uusched`s permitted on your machine.
- `Permissions` – Defines the level of access 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 needed by the `uucico` daemon, `cu`, and PPP to establish a link to a remote computer. This information includes the name of the remote host, the name of the connecting device associated with the remote host, time when the host can be reached, telephone number, login ID, and password.

Several other files may be considered part of the supporting database but are not directly involved in establishing a link and transferring files.

## *Configuring UUCP Files*

The UUCP database consists of the files shown in ”Introducing the UUCP Database Files” on page 189. However, basic UUCP configuration involves only the following critical files:

- `/etc/uucp/Systems`
- `/etc/uucp/Devices`
- `/etc/uucp/Dialers`

Because PPP uses some of the UUCP databases, you should understand at least these critical database files if you plan to configure PPP. Once these databases are configured, UUCP administration is fairly straightforward. Typically, you edit the `Systems` file first, and then edit the `Devices` file. You usually can use the default `/etc/uucp/Dialers` file, unless you plan to add dialers that aren’t in the default file. In addition, you may also want to use the following files for basic UUCP and PPP configuration:

- `/etc/uucp/Sysfiles`
- `/etc/uucp/Dialcodes`
- `/etc/uucp/Sysname`

Because these files work closely with each other, you should understand the contents of them all before you change any one of them. A change to an entry in one file may require a change to a related entry in another file. The remaining files listed in ”Introducing the UUCP Database Files” are not as critically intertwined.

---

**Note –** PPP uses only the files described in this section. It does not use the other UUCP database files.

---

The rest of this chapter explain the UUCP databases in detail.

# /etc/uucp/Systems *File*

The /etc/uucp/Systems file contains the information needed by the uucico daemon to establish a communication link to a remote computer. It is the first file 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 may have more than one entry. The additional entries represent alternative communication paths that will be 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.

Using the Sysfiles file, you can define several files to be used as Systems files. See the description of the Sysfiles file for details.

Each entry in the Systems file has the following format:

*System-Name Time Type Speed Phone Chat-Script*

Here is an example showing the fields of the Systems file.

*Table 12-1* Fields in /etc/uucp/Systems

| System-Name | Time | Type | Speed | Phone | Chat Script |
|---|---|---|---|---|---|
| arabian | Any | ACUEC | 38400 | 1112222 | Login: Puucp ssword:beledi |

## *System-Name Field*

This field contains the node name of the remote computer. On TCP/IP networks, this may be the machine's host name or a name created specifically for UUCP communications through the /etc/uucp/Sysname file. See "/etc/uucp/Sysname File" on page 214. In Table 12-1, the System-Name field contains an entry for remote host arabian.

## *Time Field*

This field specifies the day of week and time of day when the remote computer can be called. The format of the Time field is:

```
daytime[;retry]
```

The *day* portion may be a list containing some of the following entries:

*Table 12-2* Day Field

| | |
|---|---|
| `Su Mo Tu We Th Fr Sa` | For individual days |
| `Wk` | For any weekday |
| `Any` | For any day |
| `Never` | Your host will never initiate a call to the remote computer; the call must be initiated by the remote computer. Your host is then operating in *passive mode*. |

Table 12-1 shows `Any` in the Time field, indicating that host `arabian` can be called at any time.

The *time* portion should be a range of times specified in 24-hour notation. (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 Subfield*

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. What this means is that UUCP will start 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 there is no response, the next retry will be 10 minutes later. The next retry will be 20 minutes later, and so on until the maximum retry time of 23 hours is reached. If *retry* is specified, that will always be the retry time. Otherwise, the back-off algorithm is used.

## Type Field

This field contains the device type that should be used to establish the communication link to the remote computer. The keyword used in this field is matched against the first field of `Devices` file entries as shown in Table 12-3. (Note that the fields listed in the table heading are for the `Systems` file and do not apply to the `Devices` file. For a table showing the same correspondences to fields in the `Devices` file, see Table 12-8 on page 201.)

*Table 12-3* Type Field and `/etc/uucp/Devices` File

| File Name | System-Name | Time | Type | Speed | Phone | Chat-Script |
|---|---|---|---|---|---|---|
| `Systems` | `arabian` | `Any` | `ACUEC, g` | `38400` | `1112222` | `ogin: Puucp ssword:beledi.` |
| `Devices` | `ACUEC` | `cua/a` | `-` | `38400` | `usrv32bis-ec` | |

You can define the protocol used to contact the system by adding it on to the Type field. The example above shows how to attach the protocol `g` to the device type `ACUEC`. (For information on protocols, see ”Protocol Definitions in the Devices File” on page 205.)

## Speed Field

This field (also known as the Class field) specifies the transfer speed of the device used in establishing the communication link. It may contain a letter and speed (for example, `C1200`, `D1200`) to differentiate between classes of dialers (refer to ”Class Field” on page 201).

Some devices can be used at any speed, so the keyword `Any` may be used. This field must match the Class field in the associated `Devices` file entry as shown in Table 12-4:

*Table 12-4* Speed Field and `/etc/uucp/Devices` File

| File Name | SystemName | Time | Type | Speed | Phone | Chat-Script |
|---|---|---|---|---|---|---|
| `Systems` | `eagle` | `Any` | `ACU, g` | `D1200` | `NY3251` | `ogin: nuucp ssword: Oakgrass` |
| `Devices` | `ACU` | `tty11` | `--` | `D1200` | `penril` | |

If information is not required for this field, use a dash (`-`) as a place holder for the field.

## *Phone Field*

This field allows you to specify the telephone number (token) of the remote computer for automatic dialers (port selectors). The telephone number consists of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be one that is listed in the `Dialcodes` file, as shown in Table 12-5:

*Table 12-5* Phone Field Correspondence

| File Name | System-Name | Time | Type | Speed | Phone | Chat-Script |
|---|---|---|---|---|---|---|
| `Systems` | `nubian` | `Any` | `ACU` | `2400` | `NY5551212` | `ogin: Puucp ssword:Passuan` |
| `Dialcodes` | `NY 1=1212` | | | | | |

In this string, an equals sign (=) tells 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 may access other computers 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 on to the switch. In this way, the port selector will know the remote machine with which your host wishes to communicate. (This is 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 using the `Dialcodes` file.

## *Chat-Script Field*

This field (also called the Login field) contains a string of characters 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 format:

*expect send [expect send] ....*

*expect* represents the string that the local host expects to get from the remote host to initiate conversation. *send* is the string the local host will send after it 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:

- Login prompt that the local host expects to get from the remote machine
- Login name that the local host will send to the remote machine in order to log in
- Password prompt that the local host expects to get from the remote machine
- Password that the local host will send to the remote machine

The *expect* field may be made up of subfields of the form:

*expect[-send-expect]...*

where *-send* is sent if the prior *expect* is not successfully read, and *-expect* following the *send* is the next expected string.

For example, with strings `login--login`, the UUCP on the local host will expect `login`. If UUCP gets `login` from the remote machine, it will go to the next field. If it does not get `login`, it will send a carriage return, then look for `login` again. If the local computer initially does not expect any characters, use the characters `""` (NULL string) in the *expect* field. Note that all *send* fields will be sent followed by a carriage-return unless the *send* string is terminated with a `\c`.

Here 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 tells 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 get `ogin:` send the login name `Puucpx`. When you get `ssword:` (for `Password:`), send the password `xyzzy`.

Table 12-6 lists some useful escape characters.

*Table 12-6* Escape Characters Used in `Systems` File Chat Script

| | |
|---|---|
| `\b` | Send or expect a backspace character. |
| `\c` | If at the end of a string, suppress the carriage return that is normally sent. Ignored otherwise. |
| `\d` | Delay 1–3 seconds before sending more characters. |
| `\E` | Start echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.) |
| `\e` | Echo check-off. |
| `\H` | Ignore one hangup. Use this option for dialback modems. |
| `\K` | Send a BREAK character. |
| `\M` | Turn on `CLOCAL` flag. |
| `\m` | Turn off `CLOCAL` flag. |
| `\n` | Send or expect a newline character. |
| `\N` | Send a NULL character (ASCII NUL). |
| `\p` | Pause for approximately 1/4 to 1/2 second. |
| `\r` | Send or expect a carriage return. |
| `\s` | Send or expect a space character. |
| `\t` | Send or expect a tab character. |
| `EOT` | Send an EOT followed by newline twice. |
| `BREAK` | Send a break character. |
| `\`*`ddd` | Send or expect the character 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, it disconnects the link to the remote machine and then calls the remote machine back. 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 get the characters `INITIATED` from the dial-in server. After the `INITIATED` characters have been matched, the dialing facility flushes any subsequent characters it 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.

You need not have a string of characters directly preceding or following the `\H`, as shown in the sample string above.

## *Hardware Flow Control*

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`(1V) and `termio`(4) man pages for complete details.

The following example would enable hardware flow control in a `Systems` file entry:

```
unix Any ACU 2400 12015551212 "" \r login:-\r-login:-\r-login: nuucp password: xxx "" \
STTY=crtscts
```

This pseudo-send string can also be used in entries in the `Dialers` file.

### *Setting Parity*

In some cases, you will 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. For example:

```
unix Any ACU 2400 12015551212 "" P_ZERO "" \r login:-\r-login:-\r-login: nuucp password: xxx
```

In the same manner, `P_EVEN` sets parity to even (the default), `P_ODD` sets odd parity, and `P_ONE` sets the parity bit to 1.

The parity couplet can be inserted anywhere in the chat script. It applies to all information in the chat script following the `""` `P_ZERO`. It can also be used in entries in the `Dialers` file.

## `/etc/uucp/Devices` *File*

The `/etc/uucp/Devices` file contains information for all the devices that may be used to establish a link to a remote computer. These devices include ACUs—which includes modern, high-speed modems—direct links, and network connections.

Each entry in the `Devices` file has the following format:

| *Type* | *Line* | *Line2* | *Class* | *Dialer-Token-Pairs* |
|---|---|---|---|---|

Here is an entry in `/etc/uucp/Devices` for a US Robotics V.32bis modem attached to port A and running at 38,400 bps.

```
ACUEC cua/a - 38400 usrv32bis-ec
```

Each field is described below.

## *Type Field*

This field describes the type of link that the device will establish. It may contain one of the following keywords:

### `Direct` ***Keyword***

The `Direct` keyword mainly appears in entries for `cu` connections. This keyword indicates that the link is a direct link to another computer or a port selector. A separate entry should be made 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, or PPP) is made through a modem. This modem may be connected either directly to your computer or indirectly through a port selector.

### ***Port Selector***

This is a variable that is replaced in the Type field by the name of a port selector. Port selectors are devices attached to a network that prompt for the name of a calling modem and 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 ”/etc/uucp/Dialers File” on page 206 for more information.)

### *Sys-Name*

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 to an entry in `/etc/uucp/Systems` for the computer *Sys-Name*.

### *Type Field and* `/etc/uucp/Systems` *File*

Table 12-7 shows a comparison between the fields in `/etc/uucp/Devices` and fields in `/etc/uucp/Systems`. The titles of each column apply only to fields in the `Devices` file.

The keyword used in the Type field of the `Devices` file is matched against the third field of `the Systems` file entries, as indicated in the shaded fields in Table 12-7. In the `Devices` file, the Type field has the entry ACUEC, indicating an automatic call unit, in this case a V.32bis modem. This value is matched against the third field in the `Systems` file, which also contains the entry ACUEC. (See "/etc/uucp/Systems File" on page 191 for more information.)

*Table 12-7* Type Field and `/etc/uucp/Systems` File Equivalent

| | Type | Line | Line2 | Class | Dialer-Token-Pairs |
|---|---|---|---|---|---|
| `Devices` | **`ACUEC`** | `cua/a` | `-` | `38400` | `usrv32bis-ec` |
| `Systems` | `nubian` | `Any` | **`ACUEC`** | `38400` | `9998888 "" \d\d\r\n\c-ogin-\r\n\c-ogin.....` |

## *Line Field*

This field contains the device name of the line (port) associated with the `Devices` entry. For instance, if the modem associated with a particular entry was attached to the `/dev/cua/a` device (serial port A), the name entered in this field would be `cua/a`. There is an optional modem control flag, `M`, that 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*

This field is a placeholder. Always use a dash (–) here. 801 type dialers, which are not supported in the Solaris environment, use the Line2 field. Non-801 dialers will not normally use this configuration, but still require a hyphen in this field.

## *Class Field*

The Class field contains the speed of the device, if the keyword `ACU` or `Direct` is used in the Type field. However, it may contain a letter and a speed (for example, `C1200`, `D1200`) to differentiate between classes of dialers (Centrex or Dimension PBX).

This is necessary because many larger offices may have more than one type of telephone network: one network may be dedicated to serving only internal office communications while another handles the external communications. In such a case, it becomes necessary to distinguish which line(s) should be used for internal communications and which should be used for external communications.

The keyword used in the Class field of the `Devices` file is matched against the Speed field of `Systems` file as shown in Table 12-8. Note that the titles of each column apply only to fields in the `Devices` file.

*Table 12-8* Class Field and `/etc/uucp/Systems` Correspondence

| File Name | Type | Line | Line2 | Class | Dialer-Token-Pairs |
|---|---|---|---|---|---|
| `Devices` | ACU | cua/b | – | **D2400** | hayes |
| `Systems` | gobi | Any | ACU | **D2400** | 3251 ogin: nuucp ssword: taheya |

Some devices can be used at any speed, so the keyword `Any` may be used in the Class field. If `Any` is used, the line will match any speed 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

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 may be the name of a modem, a port monitor, or it may be `direct` or `uudirect` for a direct-link device. You can have any number of dialer-token pairs; if not present, it will be taken from a related entry in the `Systems` file. The *token* portion may be supplied immediately following the dialer portion.

The last dialer token pair may not be present, depending on the associated dialer. In most cases, 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 may be defined in the `Dialers` file or may 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 special dialer types include:

*Table 12-9* Dialer-Token Pairs

| | |
|---|---|
| `TCP` | TCP/IP network |
| `TLI` | Transport Level Interface Network (without STREAMS) |
| `TLIS` | Transport Level Interface Network (with STREAMS) |

See ”Protocol Definitions in the Devices File” on page 205 for more information.

### *Structure of the Dialer-Token-Pairs Field*

The DTP field may be structured four different ways, depending on the device associated with the entry:

- Directly connected modem

  If a modem is connected directly to a port on your computer, the DTP field of the associated `Devices` file entry will have 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 as shown in Table 12-10. (The titles of each column apply only to fields in the `Devices` file.)

*Table 12-10* Dialers Field and `/etc/uucp/Dialers` Correspondence

| File Name | Type | Line | Line2 | Class | Dialer-Token-Pairs |
|---|---|---|---|---|---|
| `Devices` | `ACU` | `cua/b` | `-` | `2400` | **`hayes`** |
| `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 case the phone number) is taken from the `Phone` field of a `Systems` file entry. (`\T` is implied, as described on page 205.)

- Direct link – For a direct link to a particular computer, the DTP field of the associated entry would contain the keyword `direct`. This is true for both types of direct-link entries, `Direct` and *Sys-Name* (refer to ”Type Field” on page 199).
- Computers on the same port selector – If a computer with which you wish to communicate is on the same port selector switch as your computer, your computer must first access the switch. The switch will then make 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 as shown in Table 12-11 on page 204. (The titles of each column apply only to fields in the `Devices` file.)

*Table 12-11* Dialers Field Using a Port Selector

| File Name | Type | Line | Line2 | Class | Dialer-Token-Pairs |
|---|---|---|---|---|---|
| Devices | develcon | cua/a | - | 1200 | **develcon** |
| Dialers | **develcon** | ,"" | "" | | \pr\ps\c est:\007 \E\D\e\007 |

As shown, the *token* portion is left blank. This indicates that it is retrieved from the `Systems` file. The `Systems` file entry for this computer will contain the token in the Phone field, which is normally reserved for the phone number of the computer. (Refer to "/etc/uucp/Systems File" on page 191.) This type of DTP contains an escape character (`\D`), which ensures that the contents of the Phone field will not be interpreted as a valid entry in the `Dialcodes` file.

- Modems 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 will make the connection to the modem. This type of entry requires two dialer-token-pairs. The *dialer* portion of each pair (fifth and seventh fields of entry) will be used to match entries in the `Dialers` file as shown in Table 12-10. (Note that the titles of each column apply only to fields in the `Devices` file.)

*Table 12-12* Dialers Field With Two Dialer-Token-Pairs

| File Name | Type | Line | Line2 | Class | Dialer-Token-Pairs | | |
|---|---|---|---|---|---|---|---|
| Devices | ACU | cua/b | - | 1200 | **develcon** | **vent** | **ventel** |
| Dialers | **develcon** | "" | "" | \pr\ps\c | est:\007 | \E\D\e | \007 |
| Dialers | **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 (e.g.Ventel modem) to connect to your computer. This token would be unique for each port selector since each switch may be set up differently. Once the Ventel modem has been connected, the second pair is accessed, where Ventel is the dialer and the token is retrieved from the `Systems` file.

Two escape characters may appear in a DTP field:

- `\T` – Indicates that the Phone (*token*) field should be translated using the `/etc/uucp/Dialcodes` file. This escape character is normally placed in the `/etc/uucp/Dialers` file for each caller script associated with a modem (Hayes, US Robotics, and so on). Therefore, the translation will not take place until the caller script is accessed.
- `\D` – Indicates that the Phone (*token*) field should not be translated 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 associated with network switches (`develcon` and `micom`).

## *Protocol Definitions in the* `Devices` *File*

You can define the protocol to use with each device in `/etc/uucp/Devices`. This usually is unnecessary because you can use the default or define the protocol with the particular system you are calling. (Refer to "/etc/uucp/Systems File" on page 191.) If you do specify the protocol, you must use the form:

*Type,Protocol [parameters]*

For example, you can use `TCP,te` to specify the TCP/IP protocol.

Table 12-13 shows the available protocols for the `Devices` file:

*Table 12-13* Protocols Used in `/etc/uucp/Devices`

| Protocol | Description |
|---|---|
| `t` | This protocol is commonly used for transmissions over TCP/IP and other reliable connections. It assumes error-free transmissions. |
| `g` | This is UUCP's native protocol. It 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, like TCP/IP). |
| `f` | This protocol is used for transmission over X.25 connections. It relies on flow control of the data stream, and it is meant for working over links that can (almost) be guaranteed to be error-free, specifically X.25/PAD links. A checksum is carried out over a whole file only. If a transport fails, the receiver can request retransmission(s). |

Here is an example showing a protocol designation for a device entry:

```
TCP,te - - Any TCP -
```

This example indicates that, for device `TCP`, try to use the `t` protocol. If the other end refuses, use the `e` protocol.

Note that 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.

## `/etc/uucp/Dialers` *File*

The `/etc/uucp/Dialers` file contains dialing instructions for many commonly used modems. You probably will 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 take place on a line before it can be made available for transferring data. This conversation, often referred to as a chat script, is usually a sequence of ASCII strings that is transmitted and expected, and it is often used to dial a phone number

As shown in the examples in "/etc/uucp/Devices File" on page 198 the fifth field in a `Devices` file entry is an index into the `Dialers` file or a special dialer type (`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 format:

*dialer substitutions expect-send*

Table 12-14 shows the entry for a US Robotics V.32bis modem.:

*Table 12-14* `/etc/uucp/Dialers` File Entry

| Dialer | Substitutions | Expect-Send |
|---|---|---|
| `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` |

The Dialer field matches the fifth and additional odd-numbered fields in the `Devices` file. The Substitutions field is a translate string: the first of each pair of characters is mapped to the second character in the pair. This is usually used to translate `=` and `-` into whatever the dialer requires for "wait for dial tone" and "pause."

The remaining expect-send fields are character strings.

Code Example 12-1 on page 208 shows some sample entries in the `Dialers` file as distributed when you install UUCP as part of the Solaris installation program.

*Code Example 12-1* 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

tb9600-nec =W-, "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=6S180=0\r\c OK\r \EATDT\T\r\c
CONNECT\s9600 STTY=crtscts,crtsxoff
```

Here is a list of escape characters commonly used in the send strings in the `Dialers` file:

*Table 12-15* Backslash Characters for `/etc/uucp/Dialers`

| Character | Description |
|---|---|
| `\b` | Send or expect a backspace character |
| `\c` | No newline or carriage return. |
| `\d` | Delay (approximately 2 seconds). |
| `\D` | Phone number or token without `Dialcodes` translation. |
| `\e` | Disable echo checking. |
| `\E` | Enable echo checking (for slow devices). |
| `\K` | Insert a Break character. |
| `\n` | Send newline. |
| `\nnn` | Send octal number. Additional escape characters that may be used are listed in the section "/etc/uucp/Systems File" on page 191. |
| `\N` | Send or expect a NULL character (ASCII NUL). |
| `\p` | Pause (approximately 12–14 seconds). |
| `\r` | Return. |
| `\s` | Send or expect 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 `=` will be replaced with a `W` (wait for dial tone) and any `-` with a `P` (pause).

The handshake given by the remainder of the line works as listed:

- `""` – Wait for nothing. (that is, proceed to the next thing.)
- `\d` – Delay 2 seconds, then send a carriage-return.
- `>` – Wait for a `>`.

- `Q\c` – Send a `Q` without a carriage-return.
- `:` – Expect a `:`.
- `\d-` – Delay 2 seconds, send a `-` and a carriage-return.
- `>` – Wait for a `>`.
- `s\p9\c` – Send an `s`, pause, send a `9` with no carriage-return.
- `)-W\p\r\ds\p9\c-)` – Wait for a `)`. If it is not received, process the string between the `-` characters as follows. Send a `W`, pause, send a carriage-return, delay, send an `s`, pause, send a `9`, without a carriage-return, and then wait for the `)`.
- `y\c` – Send a `y` with no carriage-return.
- `:-` Wait for a `:`.
- `\E\TP` - Enable echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.) Then, send the phone number. The `\T` means take the phone number passed as an argument and apply the `Dialcodes` translation and the modem function translation specified by field 2 of this entry. Then send a `P` and a carriage-return.
- `>` – Wait for a `>`.
- `9\c` – Send a `9` without a newline.
- `OK` – Wait for the string `OK`.

## *Hardware Flow Control*

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, and `STTY=crtscts,crtsxoff` enables both outbound and inbound hardware flow control.

`STTY` accepts all the stty modes. See the `stty`(1V) and `termio`(4) 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 some cases, you will 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
```

In the same manner, `P_EVEN` sets it to even (the default); `P_ODD` sets it to odd; and `P_ONE` sets it to one. This pseudo-send string can also be used in entries in the `Systems` file.

## *Other Basic Configuration Files*

The files in this section can be used in addition to the `Systems`, `Devices`, and `Dialers` file when doing basic UUCP configuration.

### `/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` files to provide additional information about a basic phone number that is used by several systems at the same site.

Each entry has the format:

*abbreviation dial-sequence*

where *abbreviation* represents the abbreviation used in the Phone field of the `Systems` file and *dial-sequence* represents the dial sequence passed to the dialer when that particular `Systems` file entry is accessed. Table 12-16 shows the correspondences between the two files.

*Table 12-16* Correspondences between `Dialcodes` and `Systems` Files

| | **Field Names** | | | | | |
|---|---|---|---|---|---|---|
| **Dialcodes** | **Abbreviation** | Dial-Sequence | | | | |
| **Systems** | System-Name | Time | Type | Speed | **Phone** | Chat Script |

Table 12-17 contains sample entries in a `Dialcodes` file.

*Table 12-17* 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 entry:

```
NY5551212
```

When `uucico` reads `NY` in the `Systems` file, it searches the `Dialcodes` file for `NY` and obtains the dialing sequence `1=212`. This is the dialing sequence needed for any phone call to New York City. It 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 in the `Systems` file such as `jt7867`. When `uucico` reads the entry containing `jt7867` in the `Systems` file, it would send the sequence `9=847-7867` to the dialer if the token in the dialer-token pair is `\T`.

## /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 on cu, see the cu(1c) man page.) You may want to use Sysfiles for:

- 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 may become large, making it more convenient to split it into several smaller files.

The format of the Sysfiles file is:

service=*w* systems=*x:x* dialers=*y:y* devices=*z:z*

*w* represents uucico, cu, or both 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 presented. *y* represents one or more files to be used as the Dialers file. *z* is 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.

Below is a sample /etc/uucp/Sysfiles that 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 will first look in the standard /etc/uucp/Systems. If the system they are trying to call doesn't have an entry in that file, or if the entries in the file fail, then they look in /etc/uucp/Local_Systems.

Given the above entry, cu and uucico will share the Dialers and Devices files.

When different Systems files are defined for uucico and cu services, your machine will store two different lists of Systems. You can print the uucico list using the uuname command or the cu list using the uuname -c command. Another example of the file, where the alternate files are consulted first and the default files are consulted in case of need is:

```
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
```

### /etc/uucp/Sysname *File*

Every machine that uses UUCP must have an identifying name, often referred to as the *node name*. This is the name that 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 containing the UUCP node name for your system.

## /etc/uucp/Permissions *File*

The /etc/uucp/Permissions file specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the remote computer's ability to request files and its ability to receive files queued by the local machine. Another option is available that specifies the commands that a remote machine can execute on the local computer.

## Structuring Entries

Each entry is a logical line with physical lines terminated by a backslash (\) to indicate continuation. Entries are made up of options delimited by blank space. Each option is a name-value pair in the following format:

*name=value*

*Value*s may be colon-separated lists. Note that no blank space is allowed within an option assignment.

Comment lines begin with a pound sign (#), and they occupy the entire line up to a newline character. Blank lines are ignored (even within multiple-line entries).

There are two types of `Permissions` file entries:

- `LOGNAME` – Specifies the permissions that take effect when a remote computer logs in to (calls) your computer.

---

**Note –** When a remote machine calls you, its identity is questionable unless it has a unique login and verifiable password.

---

- `MACHINE` – Specifies permissions that take effect when your computer logs in to (calls) a remote computer.

`LOGNAME` entries contain a `LOGNAME` option and `MACHINE` entries contain a `MACHINE` option. One entry may contain both options.

## Considerations

When using the `Permissions` file to restrict the level of access granted to remote computers, you should consider the following:

- All login IDs used by remote computers to log in for UUCP communications must appear in one and only one `LOGNAME` entry.
- Any site that is called whose name does not appear in a `MACHINE` entry, will have the following default permissions or restrictions:
  - Local send and receive requests will be executed.
  - The remote computer can send files to your computer's `/var/spool/uucppublic` directory.

- The commands sent by the remote computer for execution on your computer must be one of the default commands, usually `rmail`.

## REQUEST *Option*

When a remote computer calls your computer and requests to receive a file, this request can be granted or 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. This is the default value; it will be used if the `REQUEST` option is not specified. The `REQUEST` option can appear in either a `LOGNAME` (remote computer calls you) entry or a `MACHINE` (you call remote computer) entry.

## SENDFILES *Option*

When a remote computer calls your computer and completes its work, it may attempt to take work your computer has queued for it. The `SENDFILES` option specifies whether your computer can send the work queued for the remote computer.

The string `SENDFILES=yes` specifies that your computer may send the work that is queued for the remote computer as long as it 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; it is not allowed to initiate a call to this particular remote computer. (See ”/etc/uucp/Systems File” on page 191 for more information.)

The string `SENDFILES=call` specifies that files queued in your computer will be sent only when your computer calls the remote computer. The `call` value is the default for the `SENDFILE` option. This option is only significant in `LOGNAME` entries since `MACHINE` entries apply when calls are made out to remote computers. If the option is used with a `MACHINE` entry, it will be ignored.

### 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 might want to set the `MYNAME` option of the `Permissions` file. Or if you want your organization to be known as `widget` but all your modems are connected to a machine with the host name `gadget`, you can have an entry in `gadget`'s `Permissions` file that says:

```
LOGNAME=Uworld MYNAME=widget
```

Now the system `world` will 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 says:

```
MACHINE=world MYNAME=widget
```

You can also use the `MYNAME` option for testing purposes, since it allows your machine to call itself. However, since this option could be used to mask the real identity of a machine, you should use the `VALIDATE` option, as described in "VALIDATE Option" on page 221.

### 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 coming in or going out. To grant permission to deposit files in /usr/news as well as 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 was the only one specified in a WRITE option, permission to deposit files in the public directory would be denied.

You should 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.

## NOREAD *and* NOWRITE *Options*

The NOREAD and NOWRITE options specify exceptions to the READ and WRITE options or defaults. The entry:

```
READ=/ NOREAD=/etc WRITE=/var/spool/uucppublic
```

permits reading any file except those in the /etc directory (and its subdirectories—remember, these are prefixes). It 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.

## CALLBACK *Option*

You can use the CALLBACK option in LOGNAME entries to specify that no transaction will take place until the calling system is called back. There are two reasons to set up CALLBACK: 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 will be billed for the longer call.

The string CALLBACK=yes specifies that your computer must call the remote computer back before any file transfers can take place.

The default for the CALLBACK option is CALLBACK=no. If you set CALLBACK to yes, then the permissions that affect the rest of the conversation must be specified in the MACHINE entry corresponding to the caller. Do not specify these permissions in the LOGNAME, or in the LOGNAME entry that the remote machine may have set for your host.

Note that if two sites have the CALLBACK option set for each other, a conversation will never get started.

## COMMANDS *Option*

**Caution –** The COMMANDS option can compromise the security of your system. Use it 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 will generate remote execution requests and queue them to be transferred to the remote computer. Files and commands are sent to the target computer for remote execution.

This 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 it 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 entry

```
MACHINE=owl:raven:hawk:dove COMMANDS=rmail:rnews:lp
```

overrides the COMMAND default so that the computers named owl, raven, hawk, and dove can now execute rmail, rnews, and lp on your computer.

In addition to the names as specified above, there can be full path names of commands. For example,

```
COMMANDS=rmail:/usr/local/rnews:/usr/local/lp
```

specifies that command rmail uses the default search path. 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 will be executed regardless of the default path. Likewise, /usr/local/lp is the lp command that will be executed.

Including the ALL value in the list means that any command from the remote computers specified in the entry will be executed. If you use this value, you give the remote computers full access to your machine.

**Caution –** This 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.

The string

```
COMMANDS=/usr/local/rnews:ALL:/usr/local/lp
```

illustrates two points:

- The ALL value can appear anywhere in the string.
- The path names specified for rnews and lp will be 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 like cat and uucp with the COMMANDS option. Any command that reads or writes files is potentially dangerous to local security when executed by the UUCP remote execution daemon (uuxqt).

## 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 string

```
LOGNAME=Uwidget VALIDATE=widget:gadget
```

ensures that if any machine other than widget or gadget tries to log in as Uwidget, the connection is refused. 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 associated with this entry be protected. If an outsider gets that information, that particular VALIDATE option can no longer be considered secure.

Carefully consider which remote computers you will grant privileged logins andpasswords 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 LOGNAME entry

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

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. If an outsider gets the uucpfriend login and password, masquerading is easy.

But what does this have to do with the COMMANDS option, which appears only in MACHINE entries?

It links the MACHINE entry (and COMMANDS option) with a LOGNAME entry associated with a privileged login. This link is needed because the execution daemon is not running while the remote computer is logged in. In fact, it 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 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 will be 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 listed, you are really calling either eagle, owl, or hawk. Therefore, any files 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 will also be put in the privileged spool directory. You therefore have to validate that the computer has the privileged login uucpz.

## MACHINE *Entry for* OTHER

You may want to specify different option values for remote machines that are not mentioned in specific MACHINE entries. The need may 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 available for the MACHINE entry may also be set for the computers that are not mentioned in other MACHINE entries.

## *Combining* MACHINE *and* LOGNAME

You can combine MACHINE and LOGNAME entries into a single entry where the common options are the same. For example, the two entries

```
MACHINE=eagle:owl:hawk REQUEST=yes \
READ=/ WRITE=/
```

and

```
LOGNAME=uupz REQUEST=yes SENDFILES=yes \
READ=/ WRITE=/
```

share the same REQUEST, READ, and WRITE options. You can merge them, 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.

### *Forwarding*

When sending files through a series of machines, the intermediary machines must have the command `uucp` among their `COMMANDS` options. That is, if you type the command:

```
% uucp sample.txt oak\!willow\!pine\!/usr/spool/uucppublic
```

this forwarding operation will work only if machine `willow` permits `oak` to execute the program `uucp`, and if `oak` permits your machine to do the same. The machine `pine`, being the last machine designated, does not have to permit the command `uucp`. Machines are not normally set up this way.

## `/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:

*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. It 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.

## `/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 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
```

## `/etc/uucp/Grades` *File*

The `/etc/uucp/Grades` file contains the definitions for the job grades that may be used to queue jobs to a remote computer. It 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 blank space. The last field in the entry is made up of subfields also separated by spaces. If a entry takes up more than one physical line, then 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.

### *User-job-grade Field*

This field contains an administrative-defined user job grade name of up to 64 characters.

### *System-job-grade Field*

This field contains a one-character job grade to which *User-job-grade* will be 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 may be bound to more than one system job grade. It is important to note that the `Grades` file will be searched sequentially for occurrences of a user job grade. Therefore, any multiple occurrences of a system job grade should be listed according to the restriction on the maximum job size.

While there is no maximum number for the user job grades, the maximum number of system job grades 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
```

Given this configuration in a `Grades` file, these two *User-job-grade* will share the same *System-job-grade*. Since 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 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, then the built-in default grade `Z` will be used. Because the restriction field default is `Any`, multiple occurrences of the default grade are not checked.

## *Job-size Field*

This field specifies the maximum job size that can be entered in the queue. *Job-size* is measured in bytes and may be a list of the options listed in Table 12-18:

*Table 12-18* Job-size Field

| | |
|---|---|
| *nnnn* | Integer specifying the maximum job size for this job grade |
| *n*K | Decimal number representing the number of kilobytes (K is an abbreviation for kilobyte) |
| *n*M | Decimal number representing the number of megabytes (M is an abbreviation for megabyte) |
| Any | Keyword specifying that there is no maximum job size |

Here are some examples:

- 5000 represents 5000 bytes
- 10K represents 10 Kbytes
- 2M represents 2 Mbytes

## *Permit-type Field*

This field contains a keyword that denotes how to interpret the ID list. Table 12-19 lists the keywords and their meanings:

*Table 12-19* Permit-type Field

| Keyword | ID List Contents |
|---|---|
| User | Login names of users permitted to use this job grade |
| Non-user | Login names of users 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 |

### *ID-list Field*

This 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 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.

### `/etc/uucp/Devconfig` *File*

The `/etc/uucp/Devconfig` file enables you to configure devices by service—`uucp` or `cu`. `Devconfig` entries define the STREAMS modules that are used for a particular device. They have the format:

`service`=*x* `device`=*y* `push`=*z*[:*z*...]

*x* can be `cu`, `uucico`, or both 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`, and then `tirdwr`.

### `/etc/uucp/Limits` *File*

The `/etc/uucp/Limits` file controls the maximum number of simultaneous `uucico`s, `uuxqt`s, and `uusched`s that are running in the `uucp` networking. In most cases, the default values are fine and no changes are needed. If you want to change them, however, use any text editor.

The format of the `Limits` file is:

`service`=*x* `max`=*y*:

*x* can be `uucico`, `uuxqt` or `uusched`, and *y* is the limit 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 `uucico`s, five `uuxqt`s, and two `uusched`s running on your machine.

### `remote.unknown` *File*

There is one other file that affects the use of communication facilities, the `remote.unknown` file. This file is a binary program that executes when a machine not found in any of the `Systems` files starts a conversation. It will log the conversation attempt and drop the connection.

**Caution –** If you change the permissions of the `remote.unknown` file so it cannot execute, your system will accept connections from any system.

This program executes when a machine that is not in any of the `Systems` starts a conversation. It will log the conversation attempt and fail to make a connection. If you change the permissions of this file so it cannot execute (`chmod 000 remote.unknown`), your system will accept any conversation requests. This is not a trivial change, and you should have very good reasons for doing it.

## *Administrative Files*

The UUCP administrative files are described below. 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 format:

  TM.*pid.ddd*

  where *pid* is a process ID and *ddd* is a sequential three-digit number starting at 0.

  When the entire file is received, the TM.*pid.ddd* file is moved to the path name specified in the C.*sysnxxxx* file (discussed below) that caused the transmission. If processing is abnormally terminated, the TM.*pid.ddd* file may 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. Table 12-20 shows the different types of UUCP lock files.

*Table 12-20* UUCP Lock Files

| File Name | Description |
|---|---|
| LCK..*sys* | *sys* represents the name of the computer using the file |
| LCK.*dev* | *dev* represents the name of a device using the file |
| LCK.*LOG* | *LOG* represents a locked UUCP log file |

  These files may remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files will be 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 (file transfers or remote command executions) has been queued for a remote computer. The names of work files have the format:

  C.*sysnxxxx*

where *sys* is the name of the remote computer, *n* is the ASCII character representing the grade (priority) of the work, and *xxxx* is the four-digit job sequence number assigned by UUCP. Work files contain the following information:

- Full path name of the file to be sent or requested
- Full path name of the destination or user or file name
- User login name
- List of options
- Name of associated data file 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 upon 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* – Where *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 may be followed by a subsequence number, *yyy* that is used when there are several `D.` files 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 representing the grade (priority) of the work. *xxxx* is a four-digit sequence number assigned by UUCP. Execute files contain the following information:
  - Requester's login and computer name
  - Names of files 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

# 12

# Configuring and Maintaining UUCP 13

This chapter explains how to start up UUCP operations once you have modified the database file relevant to your machines. The chapter contains procedures and troubleshooting information for setting up and maintaining UUCP on machines running the Solaris environment.


| | |
|---|---|
| *Adding UUCP Logins* | *page 233* |
| *Running UUCP Over TCP/IP* | *page 236* |
| *Setting Up UUCP Security* | *page 237* |
| *Regular UUCP Maintenance* | *page 238* |
| *UUCP Error Messages* | *page 241* |


## 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.

Here is a typical entry that you might put into the `/etc/passwd` file for a remote machine permitted to access your system with a UUCP connection:

```
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, so that in some cases you may have to truncate or abbreviate it.

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.

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 comes with four shell scripts that poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. The scripts are

- `uudemon.poll`
- `uudemon.hour`
- `uudemon.admin`
- `uudemon.cleanup`

These shell scripts should execute regularly to keep UUCP running smoothly. The `crontab` file to run the scripts is automatically created in `/usr/lib/uucp/uudemon.crontab` as part of the Solaris installation process, if you select the full installation. Otherwise, it 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 1.3 93/02/02 SMI"
#
48 8,12,16 * * * /usr/libuucp/uudemon.admin
45 23 * * * /usr/lib/uucp/uudemon.cleanup
0 * * * * /usr/lib/uucp/uudemon.poll
11,41 * * * * /usr/lib/uucp/uudemon.hour
```

To activate the uudemon.crontab file, become superuser and type:

```
# su uucp
# 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, where *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 will be there when uudemon.hour is called.

## uudemon.hour *Shell Script*

The default uudemon.hour shell script:

- Calls the uusched program to search the spool directories for work files (C.) that have not been processed and 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 at the time they were transferred.

By default, uudemon.hour runs twice an hour. You may want it 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:

1. 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 listed in the lock files (/var/spool/locks).
2. Sends resulting status information to the uucp administrative login via mail.

### uudemon.cleanup *Shell Script*

The default uudemon.cleanup shell script does the following:

1. Takes log files for individual machines from the /var/uucp/.Log directory, merges them, and places them in the /var/uucp/.Old directory with other old log information.
2. 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.
3. Returns mail that cannot be delivered to the sender.
4. Mails a summary of the status information 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.

### *Activating UUCP in /etc/inetd.conf*

Make sure that the following entry in /etc/inetd.conf is not preceded by a comment mark (#):

```
uucp stream tcp nowait root /usr/sbin/in.uucpd in.uucpd
```

### *Tailoring Systems File Entries for TCP/IP*

Entries in the /etc/uucp/Systems file should have the following fields:

*System-Name Time* TCP *Port networkname Standard-Login-Cha*t

A typical entry would look like this:

```
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 is important for some sites. In the example above, the site has the UUCP node name `rochester` is different from its TCP/IP host name `ur-seneca`. Moreover, there could easily be a completely different machine running UUCP that has the TCP/IP host name of `rochester`.

The Port field in the `Systems` file should have the entry `–`. This is equivalent to listing it as `uucp`. In almost every case, the *networkname* will be the same as the system name, and the Port field will be `–`, 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 it prompts for them much as `getty` and `login` do.

### *Checking `/etc/inet/services` for UUCP*

The following entry in `/etc/inet/services` sets 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 or NIS+ as its name service, you should change the `/etc/nsswitch.conf` entry for `/etc/services` to check `files` first, and then check `nis` or `nisplus`.

## *Security, Maintenance, and Troubleshooting*

Once you have set up UUCP, maintenance is straightforward. This section explains ongoing UUCP tasks with regard 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 machine to define:

- Ways it can receive files from your machine

- Directories for which it has read and write permission
- Commands it can use for remote execution

A typical `Permissions` entry is:

```
MACHINE=datsun LOGNAME=Udatsun VALIDATE=datsun COMMANDS=rmail REQUEST=yes SENDFILES=yes
```

This entry allows files to be sent and received (to and from the “normal” UUCP directories, not from anywhere in the system) and causes the UUCP user name to be validated at login time.

## Regular UUCP Maintenance

UUCP does not require much maintenance. Apart from making sure that the `crontab` file is in place, as described in the section ”uudemon.poll Shell Script” on page 235. All you have to worry about is the growth of mail files and the public directory.

### Email for UUCP

All email messages generated by the UUCP programs and scripts go to the user ID `uucp`. If you do not log in frequently as that user, you may not realize that mail is accumulating (and consuming disk space). To solve this, make an alias in `/etc/aliases` and redirect that email either to `root` or to yourself and others responsible for maintaining UUCP. Don’t forget to run the `newaliases` command after modifying the `aliases` file.

### 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 it. However, its sticky bit is set, so its 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 make sure to clean it up periodically.

If this 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 make the mode of the directory more restrictive, perhaps just to a group of people. If you do not want to run the risk of someone filling your disk, you can even deny UUCP access to it.

## *Troubleshooting UUCP*

These procedures describe how to solve common UUCP problems.

### *Checking for Faulty Modems or ACUs*

You can check if the modems or other ACUs are not working properly in several ways.

- Run `uustat -q`. This will give counts and reasons for contact failure.
- Run `cu -d -l`*line*, where *line* is `/dev/cua/a`. This will let you 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`.)

### *Checking the /`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 things that may be out of date for a machine are its

- Phone number
- Login
- Password

### *Debugging Transmissions*

If you cannot contact a particular machine, you can check out communications to that machine with `Uutry` and `uucp`.

1. **To try to make contact, type** `/usr/lib/uucp/Uutry -r` *machine* **and press Return.**
   Replace *machine* with the host name of the machine you are having problems contacting. This command

   a. Starts the transfer daemon (`uucico`) with debugging. You will get more debugging information if you are `root`.

   b. Directs the debugging output to `/tmp/`*machine*.

   c. Prints the debugging output to your terminal (`tail -f`).

   Press Control-c to end output. You can copy the output from `/tmp/`*machine* if you want to save it.

2. **If** `Uutry` **doesn't isolate the problem, try to queue a job by typing** `uucp -r` *file machine*`\!`*/dir/file* **and press Return.**
   Replace *file* by the file you want to transfer, *machine* by the machine you want to copy to, and */dir/file* where the file will be placed on the other machine. The `-r` option queues a job but does not start the transfer.

3. **Now use** `Uutry` **again.**
   If you still cannot solve the problem, you may need to call your local support representative. Save the debugging output; it will help diagnose the problem.

You may also want to decrease or increase the level of debugging provided by `Uutry` through the `-x` *n* option, where *n* indicates the debug level. The default debug level for `Uutry` is 5.

Debug level 3 provides basic information as to when and how the connection is established, but not much information about the transmission itself. Debug level 9, on the other hand, 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, get in touch with the administrator of the other site and agree on a time for doing so.

## *Checking 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 your computer attempts to communicate with. These files contain status information on the attempted communication and whether it was successful.

### *Checking Basic Information*

There are several commands you can use to check for basic networking information.

- `uuname` – Use this command to list those machines your machine can contact.
- `uulog` – Use this command to display the contents of the log directories for particular hosts.
- `uucheck -v` – Run this command to check for the presence of files and directories needed by `uucp`. This command also checks the `Permissions` file and outputs information on the permissions you have set up.

## *UUCP Error Messages*

This section lists the error messages associated with UUCP.

### *UUCP ASSERT Error Messages*

Table 13-1 lists ASSERT error messages.

*Table 13-1* ASSERT Error Messages

| Error Message | Description/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. |

*Table 13-1* ASSERT Error Messages *(Continued)*

| Error Message | Description/Action |
|---|---|
| CAN'T ALLOCATE | A dynamic allocation failed. |
| CAN'T LOCK | An attempt to make a LCK (lock) file failed. In some cases, this is a fatal error. |
| 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 occurs when there is a problem with the sequence file access. Usually indicates a software error. |
| NO uucp SERVICE NUMBER | A TCP/IP call is attempted, but there is no entry 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. |
| BAD LINE | There is a bad line in the Devices file; there are 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. |
| RETURN FROM fixline ioctl | An ioctl(2), which should never fail, failed. There is a system driver problem. |
| BAD SPEED | A bad line speed appears in the Devices or Systems file (Class or Speed field). |

*Table 13-1* ASSERT Error Messages *(Continued)*

| Error Message | Description/Action |
|---|---|
| `BAD OPTION` | There is a bad line or option in the `Permissions` file. It must be fixed immediately. |
| `PKCGET READ` | The remote machine probably hung up. No action need be taken. |
| `PKXSTART` | The remote machine aborted in a nonrecoverable way. This can usually be ignored. |
| `TOO MANY LOCKS` | There is an internal problem. Contact your system vendor. |
| `XMV ERROR` | There is a problem with some file or directory. It is likely the spool directory, since the modes of the destinations were suppose 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 need be taken. |

## *UUCP STATUS Error Messages*

Table 13-2 is a list of the most common STATUS error messages.

*Table 13-2* UUCP STATUS Messages

| Error Message | Description/Action |
|---|---|
| OK | Status is okay. |
| NO DEVICES AVAILABLE | There is currently no device available for the call. Check to see that there is a valid device 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 given machine failed. It could be a wrong login or password, wrong number, a very slow machine, or failure in getting through the Dialer-Token-Pairs script. |
| CONVERSATION FAILED | The conversation failed after successful startup. This usually means that one side went down, the program aborted, or the line (link) was dropped. |
| DIAL FAILED | The remote machine never answered. It could be a bad dialer or the wrong phone number. |
| BAD LOGIN/MACHINE COMBINATION | The machine called us with a login/machine name that does not agree with the Permissions file. This 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 Error Messages" on page 241. |
| 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 13-2* 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 calls your machine. |
| `REMOTE HAS A LCK FILE FOR ME` | The remote machine has a `LCK` file for your machine. It could be trying to call your machine. If it has an older version of UUCP, the process that was talking to your machine may have failed, leaving the `LCK` file. If it has the new version of UUCP and is not communicating with your machine, then 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 used by your machine to login 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 may not be running a standard version of UUCP. |
| `STARTUP FAILED` | Login succeeded, but initial handshake failed. |
| `CALLER SCRIPT FAILED` | This is usually the same as `DIAL FAILED`. However, if it occurs often, suspect the caller script in the `Dialers` file. Use `Uutry` to check. |

## *UUCP Numerical Error Messages*

Table 13-3 lists the exit code numbers of error status messages produced by the `/usr/include/sysexits.h` file. Not all are currently used by `uucp`.

*Table 13-3* 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, e.g. 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 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 could also include errors like "No message" to a mailer (if it cared to catch it). |
| 67 | Address Unknown | The user specified did not exist. This might be used for mail addresses or remote logins. |
| 68 | Host Name Unknown | The host did not exist. This is used in mail addresses or network requests. |
| 69 | Service Unavailable | A service is unavailable. This can occur if a support program or file does not exist. This message also can be a catchall message when something doesn't work and you don't know why. |
| 70 | Internal Software Error | An internal software error has been detected. This should be limited to non-operating system related errors if possible. |
| 71 | System Error | An operating system error has been detected. This is intended to be used for conditions like "cannot fork","cannot create pipe." For instance, it includes `getuid` returning a user that does not exist in the `passwd` file. |
| 72 | Critical OS File Missing | Some system file like `/etc/passwd` or `/etc/utmp` does not exist, cannot be opened, or has some error such as syntax error. |

*Table 13-3* UUCP Error Messages by Number

| Message Number | Description | Meaning |
|---|---|---|
| 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, indicating something 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 is not intended for file system problems, which should use `NOINPUT` or `CANTCREAT`, but rather for higher level permissions. For example, kre uses this to restrict students who can send mail to. |
| 78 | Configuration Error | There is an error in the configuration. |
| 79 | Entry Not Found | Entry not found. |
| 79 | Maximum Listed Value | Highest value for error messages |

# 13

# *Index*

## Symbols


- (dash)
  - dial-code abbreviation, 194
  - *Line2* field placeholder, 201
  - *Speed* field placeholder, 194
* wildcard in `bootparams` database, 67
*;retry* subfield of *Time* field, 192
= dial-code abbreviation, 194
\ (escape) characters
  - `Dialers` file send strings, 209
  - `Systems` file chat script, 196


## A


`-a` option of `ifconfig` command, 93
access control, *See* `Permissions` file; security
ACK segment, 22
`ACU` keyword of *Type* field, 199
ACU, *See* Automatic Call Unit (ACU); modems
Address Resolution Protocol (ARP), 15
addresses
  - Ethernet addresses
    - described, 9
    - `ethers` database, 63, 68
  - IP, *See* IP addresses
  - loopback address, 54
  - retrieving RFCs, 25 to 26

administration, *See* network administration
administrative files (UUCP), 229 to 231
  - cleanup, 236
  - execute files (`X.`), 187, 231
  - lock files (`LCK`), 230
  - temporary data files (`TM`), 230
  - work files (`C.`), 230 to 231

administrative programs (UUCP), 187 to 188
administrative subdivisions, 38
`admintool` software manager, 132 to 133
`aliases` file, 238
`ALL` value in `COMMANDS` option, 220
anonymous FTP
  - *See also*, xxi

anonymous FTP program
  - described, 17
  - InterNIC Registration Services, 40

`anonymous` login name, 17
`Any`
  - `Grades` file keyword, 226, 227, 228
  - *Speed* field keyword, 194
  - *Time* field entry, 192

application layer
  - OSI, 13

packet life cycle
receiving host, 24
sending host, 21
TCP/IP, 16 to 19
described, 14, 16 to 17
file services, 18
name services, 18
network administration, 19
routing protocols, 19
standard TCP/IP services, 17 to 18
UNIX "r" commands, 18
ARP (Address Resolution Protocol), 15
asppp file
described, 111
starting PPP, 146
stopping PPP, 147
asppp.cf file
defaults section
dynamic-link dial-in server, 171
value definitions, 182
described, 113, 139
dynamic-link configuration, 169 to 172
editing, 145 to 146
ifconfig section
basic configuration, 140 to 141
dynamic-link dial-in server, 170 to 171
multipoint dial-in server, 143 to 144
value definitions, 181
keywords
basic configuration, 141 to 142
configuration keyword definitions, 180 to 182
dynamic-link dial-in server configuration, 171 to 172
multipoint dial-in server configuration, 144 to 145
PAP/CHAP authenticator keywords and associated strings, 176
PAP/CHAP keyword definitions, 178
PAP/CHAP keyword rules, 177
PAP/CHAP peer keywords and associated strings, 176
multipoint dial-in server, 142 to 145
obtaining diagnostic information, 155
PAP/CHAP security, 176 to 180
parts of basic file, 140 to 142
path section
basic configuration, 141 to 142
dynamic-link dial-in server, 171 to 172
multipoint dial-in server, 144 to 145
obtaining diagnostic information, 155
value definitions, 181
specifying IP addresses or host names, 125
virtual network configuration, 174 to 175
.asppp.fifo file, 114
asppp.log file
described, 114
PPP diagnostics
communications between local and remote hosts, 160 to 164
host and modem setup, 156 to 159
obtaining diagnostic information, 155
aspppd PPP link manager
described, 112
FIFO file, 114
killing and restarting, 156
verifying if PPP running, 146
aspppls PPP login service, 113, 114
ASSERT ERROR message, 244
ASSERT error messages (UUCP), 241, 241 to 243
asterisk (*) wildcard in bootparams database, 67

authenticator keywords and associated strings, 176
Automatic Call Unit (ACU)
*See also* modems; outbound communications
`Devices` file *Type* field, 199
troubleshooting, 239
UUCP hardware configuration, 186

## B

`\b` escape character
`Dialers` file, 209
`Systems` file chat script, 196
backslash (escape) characters
`Dialers` file send strings, 209
`Systems` file chat script, 196
backspace escape character, 196, 209
`BAD LINE` message, 242
`BAD LOGIN/MACHINE COMBINATION` message, 244
`BAD LOGIN_UID` message, 242
`BAD OPTION` message, 243
`BAD SPEED` message, 242
`BAD UID` message, 242
Basic Network Utilities UUCP, *See* UUCP
binary to decimal conversion, 58
BNU UUCP, *See* UUCP
booting
diskless, 49
network configuration server booting protocols, 48
processes, 77 to 79
`bootparams` database
corresponding name service files, 63
network client mode configuration, 77
overview, 67
wildcard entry, 67
bootparams protocol, 48
Break escape character, 196
`Dialers` file, 209
`Systems` file chat script, 196
BSD-based operating systems
`/etc/inet/hosts` file link, 53
`/etc/inet/netmasks` file link, 59

## C

`\c` escape character
`Dialers` file, 209
`Systems` file chat script, 196
`C.` UUCP work files
cleanup, 236
described, 230 to 231
cables (network media), 4
callback
enabling dialback through chat script, 197
`Permissions` file option, 218 to 219
`CALLBACK` option of `Permissions` file, 218 to 219
`CALLBACK REQUIRED` message, 245
`CALLER SCRIPT FAILED` message, 245
`CAN'T ACCESS DEVICE` message, 244
`CAN'T ALLOCATE` message, 242
`CAN'T CHDIR` message, 242
`CAN'T CHMOD` message, 242
`CAN'T CLOSE` message, 242
`CAN'T CREATE` message, 241
`CAN'T FORK` message, 243
`CAN'T LINK` message, 242
`CAN'T LOCK` message, 242
`CAN'T MOVE TO CORRUPTDIR` message, 242
`CAN'T OPEN` message, 241
`CAN'T READ` message, 241
`CAN'T STAT` message, 242
`CAN'T UNLINK` message, 242
`CAN'T WRITE` message, 241
carriage-return escape characters, 196, 209
Challenge-Handshake Authentication Protocol, *See* CHAP
CHAP

asppp.cf keywords
authenticator keywords and associated strings, 176
definitions, 178
peer keywords and associated strings, 176
rules, 177
described, 116
editing asppp.cf file, 176 to 180
examples, 179 to 180
installing, 176 to 177
chap_name keyword, 177, 178
chap_peer_name keyword, 177, 178
chap_peer_secret keyword, 177, 178
chap_secret keyword, 177, 178
*Chat Script* field of Systems file, 195 to 197
basic script, 195
enabling dialback, 197
escape characters, 196
*expect* field, 195
format, 195
checklist for configuring PPP, 128 to 129
Class A network numbers
addressing scheme, 33 to 34
administration, 33
described, 31
IP address space division, 33
range of numbers available, 34
Class B network numbers
addressing scheme, 33 to 34
administration, 33
described, 31 to 32
IP address space division, 33
range of numbers available, 34
Class C network numbers
addressing scheme, 33 to 34
administration, 33
described, 32
IP address space division, 33
range of numbers available, 34
*Class* field, Devices file, 201
clients, *See* diskless clients; network client mode; network clients
CLOCAL flag, turning on and off, 196
.com domain, 12
commands
execute (X.) UUCP files, 187, 231
remote execution using UUCP, 216, 219 to 222
UUCP troubleshooting, 241
COMMANDS option of Permissions file, 219 to 220, 224
VALIDATE option, 221 to 222
communications links, *See* dynamic link dial-in server; multipoint links; point-to-point links; PPP links
communications protocols, 3
*See also specific protocols*
Config file
*See also* UUCP
described, 189, 224
format, 224 to 225
configuration files
PPP (asppp.cf)
described, 113
editing, 139 to 146
specifying IP addresses or host names, 125
TCP/IP networks
/etc/defaultdomain, 52
/etc/defaultrouter, 52 to 53
/etc/hostname.*interface*, 51 to 52
/etc/nodename, 52, 75
hosts database, 53 to 56
netmasks database, 56 to 60
configuration request, 160 to 161
configuring
*See also* installing
PPP links, 131 to 147
adding security, 146
checking for errors, 147
dynamic links configuration, 165 to 172
/etc/asppp.cf file, 139 to 146
/etc/inet/hosts file, 134 to 136

/etc/passwd and /etc/shadow files, 138 to 139
installing PPP software, 132 to 133
overview, 131 to 132
sample configuration, 133 to 134
starting the PPP link, 146 to 147
stopping PPP, 147
UUCP databases, 136 to 138, 190
virtual network configuration, 172 to 175
PPP preparation, 117 to 129
checklist, 128 to 129
determining IP addressing, 123 to 126
determining requirements, 118 to 123
file space requirements, 128
hardware requirements, 127
routing considerations, 126 to 127
tasks, 117
PPP requirements, 118 to 123
dial-in server with dynamic links, 121
file space, 128
hardware, 127
hosts on a virtual network, 123
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 118 to 119
remote host-to-remote host configuration, 119 to 120
routers, 81 to 88
network interfaces, 83 to 84
overview, 82 to 83
TCP/IP configuration files, 51 to 60
/etc/defaultdomain, 52
/etc/defaultrouter, 52 to 53
/etc/hostname.*interface*, 51 to 52
/etc/nodename, 52, 75
hosts database, 53 to 56
netmasks database, 56 to 60
TCP/IP configuration modes, 47 to 50
configuration information, 47
local files mode, 47 to 48, 72 to 74
mixed configurations, 49
network client mode, 48 to 49, 75 to 76
network configuration servers, 48
sample network, 49 to 50
TCP/IP networks, 45 to 79
booting processes, 77 to 79
configuration files, 51 to 60
host configuration modes, 47 to 50
local files mode, 72 to 74
network clients, 75 to 77
network configuration parameters, 72
network configuration server setup, 74 to 75
network databases, 60 to 63, 67 to 71
nsswitch.conf file, 63 to 66
prerequisites, 46
standard TCP/IP services, 77
UUCP
adding logins, 233 to 234
database files, 136 to 138, 190
shell scripts, 234 to 236
TCP/IP networks, 236 to 237
connectivity
checking PPP connection, 151
ICMP protocol reports of failures, 15
connectors, 5
CONVERSATION FAILED message, 244
CRC (cyclical redundancy check) field, 23
crontab file (UUCP), 234 to 235
cu program
checking modems or ACUs, 239
described, 188
multiple or different configuration files, 189, 213 to 214
printing Systems lists, 214

cyclical redundancy check (CRC) field, 23

## D

\D escape character
Devices file, 205
Dialers file, 205
\d escape character
Dialers file, 209
Systems file chat script, 196
-d option of cu program, 239
.D UUCP data files
cleanup, 236
daemons
network configuration server booting protocols, 48
turning on network configuration daemons, 74 to 75
dash (-)
dial-code abbreviation, 194
*Line2* field placeholder, 201
*Speed* field placeholder, 194
data (.D) UUCP files
cleanup, 236
data communications, 19 to 24
packet life cycle, 20 to 24
packet travel through TCP/IP stack, 20
data encapsulation
*See also* packets
defined, 20
TCP/IP protocol stack and, 20 to 24
databases, *See* network databases; UUCP
datagrams
IP header, 22
IP protocol formatting, 15
packet process, 22
UDP protocol functions, 16
data-link layer
framing, 23
OSI, 13
packet life cycle
receiving host, 23
sending host, 23
TCP/IP, 14
*day* entries for *Time* field, 192
*ddd* escape character, Systems file chat script, 196
debug_level keyword, 155, 182
debugging
*See also* troubleshooting
PPP debug level, 155, 182
UUCP transmissions, 239 to 240
decimal to binary conversion, 58
default keyword of *User-job-grade* field, 226
default_route keyword, 182
defaultdomain file
deleting for network client mode, 76
described, 52
local files mode configuration, 73
defaultrouter file
automatic router protocol selection and, 85
described, 52 to 53
local files mode configuration, 73
network client mode configuration, 76
specifying router for network client, 76
defaults section of asppp.cf file
server with dynamic-links, 171
value definitions, 182
delay escape character, 196, 209
designing the network
domain name selection, 37
IP addressing scheme, 29 to 34
naming hosts, 35
overview, 2, 27 to 28
subnetting, 56 to 60
Devconfig file
*See also* UUCP
described, 189, 228
format, 228
DEVICE FAILED message, 244
DEVICE LOCKED message, 244
device transmission protocols, 205 to 206

device type for UUCP communication link, 193
Devices file, 198 to 206
*See also* UUCP
*Class* field, 201
described, 114, 189, 198
*Dialer-Token-Pairs* field, 202 to 205
editing for PPP, 137
format, 198 to 199
*Line* field, 200
*Line2* field, 201
multiple or different files, 213 to 214
PPP diagnostics, 157
protocol definitions, 205 to 206
Systems file *Speed* field and, 194
Systems file *Type* field and, 200
*Type* field, 199 to 200
diagnostics, *See* troubleshooting
DIAL FAILED message, 244
dialback
CALLBACK option of Permissions file, 218 to 219
enabling through chat script, 197
dial-code abbreviations, 189, 194
Dialcodes file
*See also* UUCP
described, 189, 211
Dialers file escape characters, 209
format, 211 to 212
sample entries, 212
Systems file and, 212
Dialers file, 206 to 211
*See also* UUCP
code example, 207 to 208
described, 114, 189, 206 to 207
Devices file *DTP* field and, 203 to 204
editing for PPP, 137
escape characters in send strings, 209
format, 207
hardware flow control, 210 to 211
multiple or different files, 213 to 214
parity setting, 211
penril entry, 209 to 210
PPP diagnostics, 157
*Dialer-Token-Pairs* field of Devices file, 202 to 205
computers on same port selector, 203 to 204
direct link, 203
directly connected modem, 202 to 203
modems connected to port selector, 204 to 205
special dialer types, 202
dial-in servers
connected to nomadic machines, 105 to 106
dynamic links
configuring, 165 to 172
described, 106 to 107
requirements, 121
/etc/passwd and /etc/shadow configuration, 138 to 139
multipoint servers
aspp.cf configuration file, 142 to 145
described, 109, 110
hosts database configuration, 135 to 136
requirements, 122
requirements, 127
UUCP, 197
dial-ins, 104
*See also* inbound communications
dial-out operations, 104
*See also* outbound communications
direct keyword of *DTP* field, 202
Direct keyword of *Type* field, 199
direct link UUCP configuration, 186
directories (UUCP)
administration, 187
error messages, 241
public directory maintenance, 238
disabling, *See* turning off
disk space requirements for PPP, 127
diskless booting, 49
diskless clients
bootparams database, 63, 67
/etc/inet/hosts file, 76

ethers database, 68
Domain Name System (DNS)
described, 18
domain name registration, 12, 39
network databases, 36, 61
selecting as name service, 36
Domain Name System(DNS)
*See also* name services
domain names
/etc/defaultdomain file, 52, 73, 76
registration, 12, 39
selecting, 37
top-level domains, 37
dotted-decimal format, 29
dropped or lost packets, 15, 91
*DTP* field, *See Dialer-Token-Pairs* field of Devices file
dynamic link dial-in server
configuring, 165 to 172
asppp.cf file, 169 to 172
/etc/passwd and /etc/shadow files, 168
hosts database, 167 to 168
IP address issues, 167
described, 106 to 107
requirements, 121
dynamic routing, 85

## E

\E escape character
Dialers file, 209
Systems file chat script, 196
\e escape character
Dialers file, 209
Systems file chat script, 196
e protocol in Devices file, 206
echo checking, 196, 209
.edu domain, 12
electronic mail, *See* e-mail
e-mail
index of RFCs, 26
InterNIC address, 39
retrieving RFCs, 25 to 26
UUCP maintenance, 238
enabling, *See* turning on
endpoint systems
*See also* PPP links
described, 103
dial-ins, 104
dial-out operations, 104
enterprise networks, 9
EOT escape character, 196
equals sign (=) in dial-code abbreviation, 194
error messages, *See* messages; troubleshooting
errors directory, 241
escape characters
Dialers file send strings, 209
Systems file chat script, 196
/etc/aliases file, 238
/etc/asppp.cf file
defaults section
dynamic-link dial-in server, 171
value definitions, 182
described, 113, 139
dynamic-link configuration, 169 to 172
editing, 145 to 146
ifconfig section
basic configuration, 140 to 141
dynamic-link dial-in server, 170 to 171
multipoint dial-in server, 143 to 144
value definitions, 181
keywords
basic configuration, 141 to 142
configuration keyword definitions, 180 to 182
dynamic-link dial-in server configuration, 171 to 172
multipoint dial-in server configuration, 144 to 145

PAP/CHAP authenticator keywords and associated strings, 176
PAP/CHAP keyword definitions, 178
PAP/CHAP keyword rules, 177
PAP/CHAP peer keywords and associated strings, 176
multipoint dial-in server, 142 to 145
obtaining diagnostic information, 155
PAP/CHAP security, 176 to 180
parts of basic file, 140 to 142
`path` section
basic configuration, 141 to 142
dynamic-link dial-in server, 171 to 172
multipoint dial-in server, 144 to 145
obtaining diagnostic information, 155
value definitions, 181
specifying IP addresses or host names, 125
virtual network configuration, 174 to 175
`/etc/bootparams` file, 67
`/etc/defaultdomain` file
deleting for network client mode, 76
described, 52
local files mode configuration, 73
`/etc/defaultrouter` file
automatic router protocol selection and, 85
described, 52 to 53
local files mode configuration, 73
network client mode configuration, 76
specifying router for network client, 76
`/etc/ethers` file, 68
`/etc/gateways` file
disabling RIP, 127
forcing machine to be a router, 85
`/etc/hostname.`*interface* file
described, 51
local files mode configuration, 73
multiple network interfaces, 51 to 52
network client mode configuration, 75
router configuration, 83
router determination at startup, 84
`/etc/hosts` file, 53
`/etc/inet/hosts` file
adding subnets, 60
editing for PPP, 134 to 136
format, 53
host name, 54
initial file, 54 to 55
local files mode configuration, 73
loopback address, 54
multiple network interfaces, 54 to 55
network client mode configuration, 75
PPP link addressing information, 124 to 125
router configuration, 84
updating for dynamic links, 167 to 168
virtual network, 173
*See also* `hosts` database
`/etc/inet/netmasks` file
adding subnets, 60
editing, 59 to 60
router configuration, 84
*See also* `netmasks` database
`/etc/inet/networks` file
overview, 69 to 70
PPP link configuration, 126
virtual network, 174
*See also* `networks` database
`/etc/inet/protocols` file, 70 to 71
*See also* `protocols` database
`/etc/inet/services` file
checking for UUCP, 237
sample, 71
*See also* `services` database
`/etc/inetd.conf` file, 236
`/etc/init.d/asppp` file

described, 111
starting PPP, 146
stopping PPP, 147
/etc/netmasks file, 59
/etc/nodename file
deleting for network client mode, 75
described, 52
/etc/nsswitch.conf file, 63 to 66
changing, 65 to 66
examples, 64
name service templates, 65
network client mode configuration, 76
syntax, 64
/etc/passwd file
dynamic link dial-in server configuration, 168
enabling UUCP logins, 233
PPP configuration, 138 to 139
virtual network configuration, 174
/etc/rc2.d/S69inet startup script, 84
/etc/shadow file
dynamic link dial-in server configuration, 168
PPP configuration, 138 to 139
virtual network configuration, 174
/etc/uucp/Config file
described, 224, 189
format, 224 to 225
*See also* UUCP
/etc/uucp/Devconfig file
described, 189, 228
format, 228
*See also* UUCP
/etc/uucp/Devices file, 198 to 206
*Class* field, 201
described, 114, 189, 198
editing for PPP, 137
format, 198 to 199
*Line* field, 200
*Line2* field, 201
multiple or different files, 213 to 214
PPP diagnostics, 157
protocol definitions, 205 to 206
Systems file *Speed* field and, 194
Systems file *Type* field and, 200
*Type* field, 199 to 200
*See also* UUCP
*Dialer-Token-Pairs* field, 202 to 205
/etc/uucp/Dialcodes file
described, 189, 211
Dialers file escape characters, 209
format, 211 to 212
sample entries, 212
Systems file and, 212
*See also* UUCP
/etc/uucp/Dialers file, 206 to 211
code example, 207 to 208
described, 114, 189, 206 to 207
Devices file *DTP* field and, 203 to 204
editing for PPP, 137
escape characters in send strings, 209
format, 207
hardware flow control, 210 to 211
multiple or different files, 213 to 214
parity setting, 211
penril entry, 209 to 210
PPP diagnostics, 157
*See also* UUCP
/etc/uucp/Grades file, 225 to 228
default grade, 226
described, 189, 225
*ID-list* field, 227, 228
*Job-size* field, 227
keywords, 226, 227
*Permit-type* field, 227
*System-job-grade* field, 225 to 226
*User-job-grade* field, 225, 226
*See also* UUCP
/etc/uucp/Limits file
described, 189, 228
format, 229
*See also* UUCP
/etc/uucp/Permissions file, 214 to 224
CALLBACK option, 218 to 219
changing node name, 217
COMMANDS option, 219 to 220, 224

considerations, 215 to 216
described, 189, 214
dialback permissions, 218 to 219
file transfer permissions, 216 to 218
format, 215
forwarding operation, 224
LOGNAME
combining with MACHINE, 223
described, 215
login IDs for remote computers, 215
MACHINE
combining with LOGNAME, 223
default permissions or restrictions, 215
described, 215
OTHER option, 223
MYNAME option, 217
NOREAD option, 218
NOWRITE option, 218
OTHER option, 223
READ option, 217 to 218
remote execution permissions, 219 to 222
REQUEST option, 216
security set up, 237
SENDFILES option, 216
structuring entries, 215
uucheck program and, 188
uuxqt daemon and, 187
VALIDATE option, 221 to 222
WRITE option, 217 to 218
*See also* UUCP
/etc/uucp/Poll file
described, 189, 224
format, 224
*See also* UUCP
/etc/uucp/Sysfiles file
described, 189, 213
format, 213
printing Systems list, 214
samples, 213 to 214
*See also* UUCP
/etc/uucp/Sysname file, 189, 214
*See also* UUCP
/etc/uucp/Systems file, 191 to 198
*Chat Script* field, 195 to 197
described, 114, 190, 191
Devices file *Class* field and, 201
Devices file *Type* field and, 200
dial-code abbreviations, 189
editing for PPP, 138
escape characters, 196
format, 191
hardware flow control, 197 to 198
multiple or different files, 189, 191, 213 to 214
parity setting, 198
*Phone* field, 194
PPP diagnostics, 157
*Speed* field, 193 to 194
*System-Name* field, 191
TCP/IP configuration, 236 to 237
*Time* field
described, 191
Never entry, 192, 216
troubleshooting, 239
*Type* field, 193
*See also* UUCP
Ethernet
addresses
described, 9
ethers database, 63, 68
network media, 4
ports, 6
ethers database
checking entries, 90
corresponding name service files, 63
network client mode configuration, 77
overview, 68
execute (X.) UUCP files
cleanup, 236
described, 231
uuxqt execution, 187
executing, *See* starting
*expect* field of *Chat Script* field, 195

## F

f protocol in Devices file, 206
FIFO file (PPP), 114
FILE EXISTS message, 242
file services, 18
file space requirements for PPP software, 128
File Transfer Protocol, *See* FTP program
file transfers (UUCP)
  *See also* Permissions file
  daemon, 186 to 187
  permissions, 216 to 218
  troubleshooting, 239 to 240
  work files C., 36 to 111, 230 to 231
flow control hardware
  Dialers file, 210 to 211
  Systems file, 197 to 198
flushing local routing tables, 153
For Your Information (FYI) documents, 25
  *See also* Requests for Comments (RFCs)
forwarding operation (UUCP), 224
Fr *Time* field entry, 192
fragmented packets, 15
framing
  data-link layer, 14, 23
  described, 23
FTP program
  anonymous FTP program described, 17
  described, 17
  InterNIC Registration Services, 40
  retrieving RFCs, 25 to 26
  *See Also* tftp program, 17
FYIs, 25
  *See also* Requests for Comments (RFCs)

## G

g protocol in Devices file, 206
gateways file
  disabling RIP, 127
  forcing machine to be a router, 85
.gov domain, 12
Grades file, 225 to 228
  *See also* UUCP
  default grade, 226
  described, 189, 225
  *ID-list* field, 227, 228
  *Job-size* field, 227
  keywords, 226, 227
  *Permit-type* field, 227
  *System-job-grade* field, 225 to 226
  *User-job-grade* field, 225, 226
Group keyword of *Permit-type* field, 227

## H

\H escape character, 196
handshake, three-way, 22
hangup, ignoring, 196
hard disk space requirements for PPP, 127
hardware
  *See also* modems
  address (Ethernet address), 8
  flow control
    Dialers file, 210 to 211
    Systems file, 197 to 198
  local area network media, 4
  network interfaces, 5
  physical layer (OSI), 13
  physical network layer (TCP/IP), 14
  PPP
    requirements, 127
    troubleshooting, 150
  serial ports, 5
  UUCP
    configurations, 186
    port selector, 199
header of packets
  described, 7
  IP header, 22
  TCP protocol functions, 16
HoneyDanBer UUCP, *See* UUCP
host configuration modes (TCP/IP), 47 to 50

local files mode, 47 to 48
mixed configurations, 49
network client mode, 48 to 49
network configuration servers, 48
sample network, 49 to 50
`hostconfig` program, 76
`hostname.`*interface* file
described, 51
local files mode configuration, 73
multiple network interfaces, 51 to 52
network client mode configuration, 75
router configuration, 83
router determination at startup, 84
hosts
booting processes, 77 to 79
checking IP connectivity, 90 to 92
described, 7
forcing to become router, 85
hardware address, 8
host name
administration, 35
creating for PPP link, 125
described, 8
`/etc/inet/hosts` file, 54
IP addresses, 8, 30
multi-homed
creating, 86 to 87
described, 7
PPP diagnostics
analyzing communications between local and remote hosts, 160 to 164
setup information, 156 to 159
receiving
defined, 7
packet travel through, 23 to 24
routing protocol selection, 85
sample network, 49 to 50
sending
defined, 7
packet travel through, 21 to 23
TCP/IP configuration modes, 47 to 50
configuration information, 47
local files mode, 47 to 48, 72 to 74
mixed configurations, 49
network client mode, 48 to 49, 75 to 76
network configuration servers, 48
sample network, 49 to 50
turning off RDISC, 87
virtual network requirements, 123
`hosts` database, 53 to 56
checking entries, 90
configuring
dial-in server, 135 to 136
remote machine, 135
corresponding name service files, 63
`/etc/inet/hosts` file
adding subnets, 60
dynamic-link dial-in server, 167 to 168
editing, 134 to 136
format, 53
host name, 54
initial file, 54 to 55
local files mode configuration, 73
loopback address, 54
multiple network interfaces, 54 to 55
network client mode configuration, 75 to 76
PPP link addressing information, 124 to 125
router configuration, 84
updating for dynamic links, 167 to 168
virtual network, 173
name service forms of, 62
name services' affect, 55 to 56
network client mode configuration, 77
host-to-host communications, 15
hyphen (-)
dial-code abbreviation, 194
*Line2* field placeholder, 201
*Speed* field placeholder, 194

## I


`-i` option
  `netstat` command, 95 to 96, 151
ICMP protocol
  described, 15
  displaying statistics, 94 to 95
  `ping` command, 91
  Router Discovery (RDISC) protocol
    automatic selection, 85
    described, 19, 82
    turning off, 87 to 88
*ID-list* field of `Grades` file, 227, 228
`ifconfig` command, 92 to 93
  checking PPP interface status, 150 to 151
  described, 92
  output, 92 to 93
  syntax, 92
`ifconfig` section of `asppp.cf` file
  basic configuration, 140 to 141
  multipoint dial-in server, 143 to 144
  server with dynamic-links, 170 to 171
  value definitions, 181
`in.rarpd` daemon, 48
`in.rdisc` program
  *See also* RDISC
  described, 82
  dynamic routing selection and, 85
  logging actions, 97
  turning off RDISC, 87 to 88
`in.routed` daemon
  *See also* RIP
  described, 82
  killing, 153
  logging actions, 97
  restarting, 153
  space-saving mode, 82, 87
  verifying if running, 153
`in.telnet` daemon, 17
`in.tftpd` daemon
  described, 48
  turning on, 74
`in.uucpd` daemon, 187
`inactivity_timeout` keyword, 142, 182
inbound communications
  callback security, 218 to 219
  dial-ins defined, 104
  enabling through UUCP chat script, 197
  overview, 115 to 116
  point-to-point links, 104
  PPP requirements
    dial-in server with dynamic links, 121
    multipoint dial-in server, 122
    network-to-network configuration, 120
    remote computer-to-network configuration, 119
    remote host-to-remote host configuration, 120
    virtual network hosts, 123
index of RFCs, 26
`inetd` daemon
  checking if running, 90
  `in.uucpd` invoked by, 187
  services started by, 77
`inetd.conf` file, 236
installing
  *See also* configuring
  PAP / CHAP, 176 to 177
  PPP software, 132 to 133
`interface` keyword
  basic configuration, 141
  defined, 181
  dynamic-link dial-in server configuration, 171
  multipoint dial-in server configuration, 144
interfaces, *See* network interfaces
Internet
  described, 9
  domain name registration, 12
  protocol suite, *See* TCP / IP protocol suite
  security information, 9

Internet Control Message Protocol, *See* ICMP protocol
Internet layer (TCP/IP)
ARP protocol, 15
described, 14, 15
ICMP protocol, 15
IP protocol, 15
packet life cycle
receiving host, 23
sending host, 22
Internet protocol suite, *See* TCP/IP protocol suite
Internet Protocol, *See* IP protocol
internetworks
defined, 40
network-to-network PPP configuration, 120
packet transfer by routers, 42 to 44
redundancy and reliability, 41
topology, 40 to 42
InterNIC
contacting, 39 to 40
IP network numbers, 12
overview, 38 to 39
Registration Services, 38 to 39
domain name registration, 39
network number assignment, 33, 39
registration services
domain name registration, 12
retrieving RFCs, 25 to 26
IP addresses
applying netmasks, 58 to 59
described, 8
designing an address scheme, 29 to 34
dotted-decimal format, 29
dynamic link dial-in server issues, 167
InterNIC network number assignment, 33, 39
IP protocol functions, 15
network classes, 30 to 34
addressing scheme, 33 to 34
Class A, 31
Class B, 31 to 32
Class C, 32
network number administration, 33
network interfaces and, 34
parts, 29 to 30
host part, 30
network part, 30
subnet number, 30
PPP link
creating unique IP address and host name, 125
network number assignment, 126
requirements, 118 to 123
specifying addresses, 123 to 124
types of schemes, 124 to 126
using primary network interface IP address, 124 to 125
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 118
remote host-to-remote host configuration, 119
virtual network hosts, 123
range of numbers available, 34
space division by network class, 33
subnet issues, 56 to 59
symbolic names for network numbers, 60
virtual network issues, 173
IP datagrams
IP header, 22
IP protocol formatting, 15
packet process, 22
UDP protocol functions, 16
IP network numbers, 12
IP protocol
*See also* TCP/IP protocol suite
checking host connectivity, 90 to 92

described, 15
displaying statistics, 94 to 95
IP routing table status, 96 to 97
ipcp_async_map keyword, 182
ipcp_compression keyword, 182
ipdn virtual network interface, 103
ipdptpn virtual network interface, 103


## J


job grades (UUCP), *See* Grades file
*Job-size* field of Grades file, 227


## K


\K escape character
Dialers file, 209
Systems file chat script, 196
keywords
asppp.cf file
basic configuration, 141 to 142
configuration keyword definitions, 180 to 182
dynamic-link dial-in server configuration, 171 to 172
multipoint dial-in server configuration, 144 to 145
PAP/CHAP, 176 to 178
Devices file *Type* field, 199 to 200
Grades file, 226, 227, 228
killing
*See also* stopping
aspppd daemon, 156
in.routed daemon, 153


## L


-l option of cu program, 239
LAN, *See* local-area network (LAN)
LCK UUCP lock files, 230
lcp_compression keyword, 182
lcp_mru keyword, 182
Limits file
*See also* UUCP
described, 189, 228
format, 229
*Line* field of Devices file, 200
*Line2* field of Devices file, 201
link manager (aspppd)
described, 112
FIFO file, 114
killing and restarting, 156
verifying if PPP running, 146
links, *See* multipoint links; PPP links
loading, *See* starting
local area network (LAN)
boot processes, 77 to 79
hardware
network interfaces, 5
network media, 4
serial ports, 5
IP addresses, 30
software transfer of information, 6 to 9
hosts, 7 to 9
packets, 7
UUCP configuration, 186
WAN access, 9
security issues, 9
local files mode
defined, 47
host configuration, 72 to 74
machines requiring, 47 to 48
network configuration servers, 48
local files name service
*See also* network databases
described, 37
/etc/inet/hosts file
example, 55 to 56
format, 53
initial file, 54 to 55
requirements, 55
local files mode, 47 to 48
network databases, 61
lock (LCK) UUCP files, 230
logging
displaying UUCP log files, 187
in.rdisc program actions, 97

in.routed daemon actions, 97
PPP log file, 114
UUCP log file cleanup, 236
LOGIN FAILED message, 244
Login field, *See Chat Script* field of Systems file
login service (PPP), 113, 114
logins (UUCP)
adding, 233 to 234
privileged, 221 to 222
LOGNAME Permissions file
*See also* Permissions file
combining with MACHINE, 223
described, 215
login IDs for remote computers, 215
SENDFILES option, 216
VALIDATE option, 221 to 222
loopback address, 54
lost or dropped packets, 15, 91


## M


\M escape character, 196
\m escape character, 196
MACHINE Permissions file
*See also* Permissions file
combining with LOGNAME, 223
COMMANDS option, 219 to 220
default permissions or restrictions, 215
described, 215
OTHER option, 223
maintaining UUCP
adding logins, 233 to 234
mail, 238
public directory, 238
regular maintenance, 238 to 239
shell scripts, 234 to 236
media, network, 4, 9
message of packets
described, 7
displaying contents, 97
messages
*See also* troubleshooting
Permissions denied, 153
PPP diagnostics
communications between local and remote hosts, 160 to 164
host and modem setup, 156 to 159
UUCP
ASSERT error messages, 241 to 243
checking error messages, 240
STATUS error messages, 244 to 245
Mo *Time* field entry, 192
modems
*See also* Automatic Call Unit (ACU); outbound communications
direct connection, 202 to 203
port selector connection, 204 to 205
PPP diagnostics, 156 to 159
PPP requirements, 127
serial ports, 5
setting characteristics, 197 to 198, 210 to 211
UUCP databases
described, 114
*DTP* field of Devices file, 202 to 203, 204 to 205
editing, 136 to 138
UUCP hardware configuration, 186
UUCP troubleshooting, 239
multi-homed hosts
*See also* multiple network interfaces
creating, 86 to 87
described, 7
multiple network interfaces
/etc/hostname.*interface* file, 51 to 52
/etc/inet/hosts file, 54 to 55
router configuration, 83 to 84
multiple routers, 76
multipoint links
*See also* PPP links
described, 109
dial-in servers

aspp.cf configuration file, 142 to 145
described, 109, 110
hosts database configuration, 135 to 136
requirements, 122
requirements, 122 to 123
security, 116
virtual networks
described, 110 to 111
interface support, 103
network number assignment, 126
requirements, 123
MYNAME option of Permissions file, 217

## N

\N escape character
Dialers file, 209
Systems file chat script, 196
\n escape character
Dialers file, 209
Systems file chat script, 196
name services
*See also specific name services*
administrative subdivisions, 38
database search order specification, 63 to 66
domain name registration, 12, 39
Domain Name System (DNS), 18, 36
files corresponding to network databases, 63
hosts database and, 55 to 56
local files
described, 37
/etc/inet/hosts file, 53 to 56
local files mode, 47 to 48
network databases and, 36, 61
NIS, 36
NIS+, 18, 36
nsswitch.conf file templates, 65
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 119
remote host-to-remote host configuration, 119
virtual network hosts, 123
selecting a service, 36 to 38
supported services, 36
names/naming
*See also* addresses; name services
domain names
registration, 12, 39
selecting, 37
top-level domains, 37
host name
administration, 35
creating for PPP link, 125
described, 8
/etc/inet/hosts file, 54
naming network entities, 35 to 38
node name
local host, 52, 75
UUCP alias, 189, 217
UUCP remote computer, 191, 214
negotiate_address keyword, 182
netmasks database, 56 to 60
adding subnets, 60
corresponding name service files, 63
/etc/inet/netmasks file
adding subnets, 60
editing, 59 to 60
router configuration, 84
network masks
applying to IP address, 58 to 59
creating, 57 to 59
described, 57
subnetting, 56
netstat command
checking local routing tables, 152 to 153
checking PPP interface activity, 151
described, 93

network interface status display, 95 to 96
per protocol statistics display, 94 to 95
routing table status display, 96 to 97
running software checks, 90
syntax, 93
network administration
host names, 35
network administrator responsibilities
designing the network, 2, 27 to 28
expanding the network, 3
maintaining the network, 2 to 3
overview, 1 to 2
setting up the network, 2
network numbers, 33
Simple Network Management Protocol (SNMP), 19
network classes, 30 to 34
addressing scheme, 33 to 34
Class A, 31
Class B, 31 to 32
Class C, 32
InterNIC network number assignment, 33, 39
IP address space division, 33
network number administration, 33
range of numbers available, 34
network client mode
defined, 47
host configuration, 75 to 76
overview, 48 to 49
network clients
after installing, 77
diskless booting, 49
`ethers` database, 68
host configuration, 75 to 76
machines operating as, 48 to 49
network configuration server for, 48, 74 to 75
router specification, 76
network configuration servers
*See also* network client mode; network clients
booting protocols, 48
defined, 48
setting up, 74 to 75
network databases, 60 to 63, 67 to 71
*See also* `hosts` database
`bootparams`
network client mode configuration, 77
overview, 67
corresponding name service files, 63
DNS boot and data files and, 61
`ethers`
checking entries, 90
network client mode configuration, 77
overview, 68
`hosts`
*See also* `/etc/inet/hosts` file
checking entries, 90
configuring, 135 to 136
name service forms of, 62
name services' affect, 55 to 56
network client mode configuration, 77
overview, 53 to 56
name services' affect, 61 to 63
`netmasks`, 56 to 60, 63
`networks`
overview, 69 to 70
PPP link configuration, 126
virtual network, 174
`nsswitch.conf` file and, 61, 63 to 66
`protocols`, 70 to 71
`services`
overview, 71
UUCP port, 237
network interfaces
checking PPP interface activity, 151
status, 150 to 151
described, 5
displaying configuration information, 92 to 93

displaying status, 95 to 96
IP addresses and, 34
multiple network interfaces
/etc/hostname.*interface* file, 51 to 52
/etc/inet/hosts file, 54 to 55
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 118
remote host-to-remote host configuration, 119
virtual network hosts, 123
PPP virtual network interfaces, 103
primary
defined, 6
host name and, 8
using IP address for PPP link, 124 to 125
router configuration, 83 to 84
network layer (OSI), 13
network media, 4, 9
network numbers, *See* IP address; network classes
network planning, 27 to 44
adding routers, 40 to 44
design decisions, 27 to 28
IP addressing scheme, 29 to 34
name assignments, 35 to 38
registering your network, 38 to 40
software factors, 28
network topology, 40 to 42
networks database
corresponding name service files, 63
overview, 69 to 70
PPP link configuration, 126
virtual network, 174
networks, *See* local-area network (LAN); PPP links; TCP/IP networks; wide-area network (WAN)
network-to-network PPP configuration requirements, 120
Never *Time* field entry, 192, 216
newaliases command, 238
newline escape characters, 196, 209
NFS service, 18
NIS
*See also* name services
domain name registration, 12, 39
network databases, 36, 61
selecting as name service, 36
NIS+
*See also* name services
described, 18
domain name registration, 12, 39
network databases, 36, 61
selecting as name service, 36
*nnn* escape character, 209
NO DEVICES AVAILABLE message, 244
NO UUCP SERVICE NUMBERmessage, 242
node name
local host, 52, 75
UUCP alias, 189, 217
UUCP remote computer, 191, 214
nodename file
deleting for network client mode, 75
described, 52
nomadic machines connected to dial-in server, 105 to 106
Non-group keyword of *Permit-type* field, 227
Non-user keyword of *Permit-type* field, 227
NOREAD option of Permissions file, 218
NOWRITE option of Permissions file, 218
nsswitch.conf file, 63 to 66
changing, 65 to 66
examples, 64
name service templates, 65
network client mode configuration, 76

syntax, 64
NUL (ASCII character) escape character, 196, 209
null escape character, 196, 209

## O

octal numbers escape character, 196, 209
OK message, 244
Open Systems Interconnect (OSI) Reference Model, 12 to 13
opening, *See* starting
OTHER option of Permissions file, 223
outbound communications
dial-out operations, 104
editing UUCP databases, 136 to 138
overview, 114 to 115
point-to-point links, 104
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 119
remote host-to-remote host configuration, 120
virtual network hosts, 123

## P

\p escape character
Dialers file, 209
Systems file chat script, 196
packets
checking flow, 153 to 154
data encapsulation, 21 to 22
described, 7, 20
displaying contents, 97
dropped or lost, 15, 91
fragmentation, 15
header
described, 7
IP header, 22
TCP protocol functions, 16
IP protocol functions, 15
life cycle, 20 to 24
application layer, 21
data-link layer, 23
Internet layer, 22
physical network layer, 23
receiving host process, 23 to 24
transport layer, 21 to 22
message, 7
PPP
configuration request, 160 to 161
diagnostics, 160 to 164
process in TCP/IP stack, 20
transfer
router, 42 to 44
TCP/IP stack, 20 to 24
transfer log, 97
UDP, 22
PAP
asppp.cf keywords
authenticator keywords and associated strings, 176
definitions, 178
peer keywords and associated strings, 176
rules, 177
described, 116
editing asppp.cf file, 176 to 180
examples, 179 to 180
installing, 176 to 177
pap_id keyword
associated string, 176
defined, 178
value if not specified, 177
pap_password keyword
associated string, 176
defined, 178
value if not specified, 177
pap_peer_id keyword
associated string, 176
defined, 178
value if not specified, 177
pap_peer_password keyword
associated string, 176

defined, 178
value if not specified, 177
parity
Dialers file, 211
Systems file, 198
passive mode, 216
passwd file
dynamic link dial-in server configuration, 168
enabling UUCP logins, 233
PPP configuration, 138 to 139
virtual network configuration, 174
Password Authentication Protocol, *See* PAP
passwords (UUCP), privileged, 221 to 222
path section of asppp.cf file
basic configuration, 141 to 142
dynamic-link dial-in server, 171 to 172
multipoint dial-in server, 144 to 145
obtaining diagnostic information, 155
value definitions, 181
peer keywords and associated strings, 176
peer_ip_address keyword
defined, 181, 182
dynamic-link dial-in server configuration, 171
multipoint dial-in server configuration, 145
peer_system_name keyword
basic configuration, 142
defined, 181
dynamic-link dial-in server configuration, 171
multipoint dial-in server configuration, 144 to 145
penril entry in Dialers file, 209 to 210
Permissions denied message, 153
Permissions file, 214 to 224
*See also* UUCP
CALLBACK option, 218 to 219
changing node name, 217
COMMANDS option, 219 to 220, 224
considerations, 215 to 216
described, 189, 214
dialback permissions, 218 to 219
file transfer permissions, 216 to 218
format, 215
forwarding operation, 224
LOGNAME
combining with MACHINE, 223
described, 215
login IDs for remote computers, 215
MACHINE
combining with LOGNAME, 223
default permissions or restrictions, 215
described, 215
OTHER option, 223
MYNAME option, 217
NOREAD option, 218
NOWRITE option, 218
OTHER option, 223
READ option, 217 to 218
remote execution permissions, 219 to 222
REQUEST option, 216
security set up, 237
SENDFILES option, 216
structuring entries, 215
uucheck program and, 188
uuxqt daemon and, 187
VALIDATE option, 221 to 222
WRITE option, 217
*Permit-type* field of Grades file, 227
*Phone* field of Systems file, 194
physical layer (OSI), 13
physical network layer (TCP/IP), 14, 23
ping command, 90 to 92
checking PPP connection, 151
described, 90
running, 91 to 92
syntax, 90 to 91
verifying if PPP running, 147
PKCGET READ message, 243
pkgadd program, 132 to 133

pkginfo command, 132
PKXSTART message, 243
planning your network, *See* network planning
plumb option of ifconfig, 140
point-to-multipoint links, *See* multipoint links
point-to-point links
  *See also* PPP links
  communications links defined, 103
  described, 103
  dial-in server with dynamic link
    configuring, 165 to 172
    described, 106 to 107
    requirements, 121
  dial-ins and inbound communications, 104
  dial-out operations and outbound communications, 104
  generic configuration, 103 to 104
  nomadic machines connected to dial-in server, 105 to 106
  requirements, 118 to 121
  security, 116
  two isolated hosts connected, 105
  two networks connected, 107 to 108
Point-to-Point Protocol, *See* PPP links; PPP protocol
Poll file
  *See also* UUCP
  described, 189, 224
  format, 224
polling remote computers (UUCP), 189, 224
*Port Selector* variable in Devices file, 199
ports
  Devices file entry, 200
  Ethernet ports, 6
  serial ports
    described, 5
    PPP transmission facilities, 102
    selection for PPP, 127
  TCP and UDP port numbers, 71
  UUCP, 237
PPP links
  communications links defined, 103
  configuration preparation, 117 to 129
    checklist, 128 to 129
    determining IP addressing, 123 to 126
    determining requirements, 118 to 123
    file space requirements, 128
    hardware requirements, 127
    routing considerations, 126 to 127
    tasks, 117
  configuration request packet, 160 to 161
  configurations supported
    multipoint, 109 to 111
    point-to-point, 105 to 108
  configuring, 131 to 147, 165 to 182
    adding security, 146
    asppp.cf keywords, 180 to 182
    checking for errors, 147
    dynamic links configuration, 165 to 172
    /etc/asppp.cf file, 139 to 146
    /etc/inet/hosts file, 134 to 136
    /etc/passwd and /etc/shadow files, 138 to 139
    installing PPP software, 132 to 133
    overview, 131 to 132
    sample configuration, 133 to 134
    security, 176 to 180
    starting the PPP link, 146 to 147
    stopping PPP links, 147
    UUCP databases, 136 to 138, 190
    virtual network configuration, 172 to 175
  diagnostics, 155 to 164
    analyzing diagnostic output, 156 to 164
    communications between local and remote hosts, 160 to 164

described, 155
editing `asppp.cf` file, 155
host and modem setup, 156 to 159
obtaining diagnostic information, 155
setting diagnostics for your machine, 155
forcing machine to be a router, 85
IP addresses, 123 to 126
creating unique address and host name, 125
network number assignment, 126
specifying, 123 to 124
types of schemes, 124 to 126
using primary network interface address, 124 to 125
multipoint links
described, 109
dial-in servers, 109, 110, 122
virtual networks, 110 to 111, 123
point-to-point links
described, 103
dial-in server with dynamic link, 106 to 107, 121, 165 to 172
dial-ins and inbound communications, 104
dial-out operations and outbound communications, 104
generic configuration, 103 to 104
network-to-network, 107 to 108, 120
nomadic machines connected to dial-in server, 105 to 106
remote host-to-remote host, 119 to 120
two isolated hosts connected, 105
requirements, 118 to 123
dial-in server with dynamic links, 121
file space, 128
hardware, 127
hosts on a virtual network, 123
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 118 to 119
remote host-to-remote host configuration, 119 to 120
run-control script, 111
security, 116, 146, 176 to 180
starting, 146 to 147
stopping, 147
troubleshooting, 149 to 164
connectivity, 151
diagnostics, 155 to 164
hardware, 150
interface activity, 151
interface status, 150 to 151
local routing tables, 152 to 153
order for checks, 149 to 150
packet flow, 153 to 154
permissions, 153
verifying if running, 146 to 147
virtual network interface support, 103
PPP protocol
configurations supported
multipoint, 109 to 111
point-to-point, 105 to 108
overview, 101 to 103
run-control script, 111
security, 116, 146, 176 to 180
software components, 111 to 112
configuration file, 113
FIFO file, 114
file space requirements, 128
inbound connections scenario, 115 to 116
installing, 132
link manager, 112
log file, 114
login service, 113
outbound connections scenario, 114 to 115
UUCP databases, 114

verifying installation, 132 to 133
Solaris configurations supported, 105 to 111
Solaris specifications, 102
standards conformance, 102 to 103
starting the PPP link, 146 to 147
stopping, 147
transmission facilities supported, 102
verifying if running, 146 to 147
virtual network interfaces, 103
presentation layer (OSI), 13
primary network interface
defined, 6
host name and, 8
using IP address for PPP link, 124 to 125
problem solving, *See* troubleshooting
protocol definitions in `Devices` file, 205 to 206
protocol layers
*See also specific layers*
OSI Reference Model, 12 to 13
packet life cycle, 20 to 24
packet travel through, 20
TCP/IP protocol architecture model, 13 to 19
application layer, 14, 16 to 19
data-link layer, 14
Internet layer, 14, 15
physical network layer, 14
transport layer, 14, 16
protocol stacks, *See* protocol layers
protocol statistics display, 94 to 95
`protocols` database
corresponding name service files, 63
overview, 70 to 71
`ps` command, 146, 153
public directory maintenance (UUCP), 238

## Q

`-q` option
`in.routed` daemon, 82
`uustat` program, 239
queue (UUCP)
administrative files, 229 to 231
clean-up program, 188
job grade definitions, 225 to 228
scheduling daemon, 187
spool directory, 230
`uusched` daemon
described, 187
maximum simultaneous executions, 189, 228 to 229

## R

"r" commands, 18
`\r` escape character
`Dialers` file, 209
`Systems` file chat script, 196
`-r` option
`netstat` command, 96 to 97, 152
`uucp` program, 240
`Uutry` program, 240
RARP protocol
checking Ethernet addresses, 90
described, 48
Ethernet address mapping, 68
RARP server configuration, 74 to 75
RDISC
*See also* `in.rdisc` program
automatic selection, 85
described, 19, 82
turning off, 87 to 88
`READ` option of `Permissions` file, 217 to 218
`NOREAD` option, 218
receiving hosts
defined, 7
packet travel through, 23 to 24
redirection of ICMP protocol reports, 15
registering
domain names, 12, 39
networks, 38 to 40

remote computer-to-network PPP configuration, 118 to 119
REMOTE DOES NOT KNOW ME message, 245
remote execution (UUCP)
commands, 216, 219 to 222
daemon, 187
work files C., 230 to 231
REMOTE HAS A LCK FILE FOR ME message, 245
remote host-to-remote host PPP configuration, 119 to 120
REMOTE REJECT AFTER LOGIN message, 245
REMOTE REJECT, UNKNOWN MESSAGE message, 245
remote.unknown file, 229
REQUEST option of Permissions file, 216
Requests for Comments (RFCs)
described, 24 to 25
FYIs, 25
obtaining, 25 to 26
online index, 26
paper copies, 26
require_authentication keyword
associated string, 176
defined, 178
syntax, 177
requirements
PPP, 118 to 123
dial-in server with dynamic links, 121
file space, 128
hardware, 127
hosts on a virtual network, 123
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 118 to 119
remote host-to-remote host configuration, 119 to 120
restarting, *See* starting
*;retry* subfield of *Time* field, 192
return escape character, 209
RETURN FROM fixline ioctl message, 242
Reverse Address Resolution Protocol, *See* RARP protocol
RFCs, *See* Requests for Comments (RFCs)
RIP
*See also* in.routed daemon
automatic selection, 85
described, 19, 82
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 119
remote host-to-remote host configuration, 120
virtual network hosts, 123
starting for multipoint links, 127
turning off, 126 to 127
rlogin command
packet process, 21
route command, 153
Router Discovery (RDISC) protocol, *See* RDISC
routers
*See also* multiple network interfaces
adding, 40 to 44
configuring, 81 to 88
network interfaces, 83 to 84
overview, 82 to 83
default address, 72
defined, 81
described, 7
determining if a machine is a router, 84
dynamic vs. static routing, 85
/etc/defaultrouter file, 52 to 53
forcing machines as, 85
local files mode configuration, 73

network client specification, 76
network topology, 40 to 42
packet transfer, 42 to 44
routing protocols
automatic selection, 85
described, 19, 81 to 82
PPP requirements, 119 to 123
turning off RDISC, 87 to 88
turning off RIP, 127
Routing Information Protocol, *See* RIP
routing protocols
automatic selection, 85
described, 19, 81 to 82
PPP requirements
dial-in server with dynamic links, 121
multipoint dial-in server, 122
network-to-network configuration, 120
remote computer-to-network configuration, 119
remote host-to-remote host configuration, 120
turning off RIP, 127
virtual network hosts, 123
RDISC
automatic selection, 85
described, 19, 82
turning off, 87 to 88
RIP
automatic selection, 85
described, 19, 82
PPP requirements, 119 to 123
starting for multipoint links, 127
turning off, 126 to 127
routing tables
checking local tables, 152 to 153
described, 42
displaying, 90
flushing, 153
`in.routd` daemon creation of, 82
IP routing table status, 96 to 97
packet transfer example, 43 to 44
space-saving mode, 82, 87
subnetting and, 56
`rpc.bootparamd` daemon, 48
RS-232 telephone lines
PPP requirement, 127
UUCP configuration, 186
running, *See* starting


## S


`\s` escape character
`Dialers` file, 209
`Systems` file chat script, 196
`-S` option of `in.routed` daemon, 82, 87
`-s` option
`netstat` command, 94 to 95
`ping` command, 91 to 92
`s69inet` startup script, 84
`Sa` *Time* field entry, 192
scheduling daemon for UUCP, 187
scripts
chat scripts (UUCP), 195 to 197
basic script, 195
enabling dialback, 197
escape characters, 196
*expect* field, 195
format, 195
PPP run-control script, 111
shell scripts (UUCP), 234 to 236
startup scripts, 78, 84
security
checking permissions, 153
Internet information source, 9
PPP, 116, 146
UUCP
*See also* `Permissions` file
`COMMANDS` option of `Permissions` file, 219 to 220
setting up, 237
sticky bit for public directory files, 238
`VALIDATE` option of `Permissions` file, 221 to 222
WAN access issues, 9

SENDFILES option of Permissions file, 216
sending hosts
defined, 7
packet travel through, 21 to 23
serial ports
described, 5
PPP transmission facilities, 102
selection for PPP, 127
servers, dial-in, *See* dial-in servers
services database
corresponding name service files, 63
overview, 71
UUCP port, 237
session layer (OSI), 13
shadow file
dynamic link dial-in server configuration, 168
PPP configuration, 138 to 139
virtual network configuration, 174
shell scripts (UUCP), 234 to 236
automatic execution, 234
running manually, 234 to 235
uudemon.admin, 235
uudemon.cleanup, 236
uudemon.hour
described, 235
uusched daemon execution by, 187
uuxqt daemon execution by, 187
uudemon.poll, 224, 235
Simple Network Management Protocol (SNMP), 19
SNMP (Simple Network Management Protocol), 19
snoop command
checking packet flow, 153 to 154
displaying packet contents, 97
software checks (TCP/IP), 90
Solaris
LAN hardware
network interfaces, 5
network media, 4
serial ports, 5
PPP
configurations supported, 105 to 111
specifications, 102
UUCP version, 185
space escape character, 196, 209
space-saving mode
in.routed daemon option, 82
turning on, 87
*Speed* field
Devices file *Class* field and, 201
Systems file, 193 to 194
spool (UUCP)
administrative files, 229 to 231
clean-up program, 188
directory, 230
job grade definitions, 225 to 228
uusched daemon
described, 187
maximum simultaneous executions, 189, 228 to 229
starting
*See also* turning on
booting
diskless, 49
network configuration server booting protocols, 48
processes, 77 to 79
enabling dialback through chat script, 197
killing and restarting aspppd daemon, 156
PPP link, 146 to 147
restarting in.routed daemon, 153
startup scripts, 78, 84
turning on
CLOCAL flag, 196
echo checking, 196, 209
network configuration daemons, 74 to 75
space-saving mode, 87
UUCP shell scripts, 234 to 236
STARTUP FAILED message, 245

startup scripts, 78, 84
static routing, 85
statistics
IP routing table status, 96 to 97
network interface (ifconfig), 93
packet transmission (ping), 91 to 92
per-protocol (netstat), 94 to 95
PPP interface, 150 to 152
.Status directory, 241
STATUS error messages (UUCP), 241, 244 to 245
sticky bit for public directory files, 238
stopping
*See also* turning off
killing and restarting aspppd daemon, 156
killing in.routed daemon, 153
PPP, 147
turning off
CLOCAL flag, 196
echo checking, 196, 209
RDISC, 87 to 88
RIP, 126 to 127
STREAMS
device configuration, 228
dialer token pairs, 202
STTY flow control, 197 to 198, 210 to 211
Su *Time* field entry, 192
subdivisions, administrative, 38
subnetting
adding subnets, 60
IP addresses and, 57 to 59
local files mode configuration, 74
netmasks database, 56 to 60
editing /etc/inet/netmasks file, 59 to 60
network mask creation, 57 to 59
network configuration servers, 48
network masks
applying to IP address, 58 to 59
creating, 57 to 59
described, 57
overview, 56
subnet number in IP addresses, 30
subscribing to Internet security information, 9
symbolic names for network numbers, 60
SYN segment, 22
Sysfiles file
*See also* UUCP
described, 189, 213
format, 213
printing Systems list, 214
samples, 213 to 214
SYSLST OVERFLOW message, 242
Sysname file, 189, 214
*See also* UUCP
*Sys-Name* variable of *Type* field, 200
SYSTEM NOT IN Systems FILE message, 244
*System-job-grade* field of Grades file, 225 to 226
*System-Name* field of Systems file, 191
Systems file, 191 to 198
*See also* UUCP
*Chat Script* field, 195 to 197
described, 114, 190, 191
Devices file *Class* field and, 201
Devices file *Type* field and, 200
dial-code abbreviations, 189, 194
editing for PPP, 138
escape characters, 196
format, 191
hardware flow control, 197 to 198
multiple or different files, 189, 191, 213 to 214
parity setting, 198
*Phone* field, 194
PPP diagnostics, 157
*Speed* field, 193 to 194
*System-Name* field, 191
TCP/IP configuration, 236 to 237
*Time* field
described, 191
Never entry, 192, 216
troubleshooting, 239
*Type* field, 193

## T


\T escape character
    Devices file, 205
    Dialers file, 205, 209
\t escape character, 196
t protocol in Devices file, 206
-t option of inetd daemon, 77
tab escape character, 196
TALKING message, 244
TCP connection-tracing, 77
TCP dialer type, 202
TCP protocol
    *See also* TCP/IP protocol suite
    described, 16
    displaying statistics, 94 to 95
    establishing a connection, 22
    segmentation, 21
    services in /etc/inet/services file, 71
TCP/IP networks
    configuration files, 51 to 60
        /etc/defaultdomain, 52
        /etc/defaultrouter, 52 to 53
        /etc/hostname.*interface*, 51 to 52
        /etc/nodename, 52, 75
        hosts database, 53 to 56
        netmasks database, 56 to 60
    configuring, 45 to 79
        booting processes, 77 to 79
        configuration files, 51 to 60
        host configuration modes, 47 to 50
        local files mode, 72 to 74
        network clients, 75 to 77
        network configuration parameters, 72
        network configuration server setup, 74 to 75
        network databases, 60 to 63, 67 to 71
        nsswitch.conf file, 63 to 66
        prerequisites, 46
        standard TCP/IP services, 77
    host configuration modes, 47 to 50
        local files mode, 47 to 48
        mixed configurations, 49
        network client mode, 48 to 49
        network configuration servers, 48
        sample network, 49 to 50
    IP network numbers, 12
    troubleshooting, 89 to 97
        displaying packet contents, 97
        general methods, 89 to 90
        ifconfig command, 92 to 93
        logging routing daemon actions, 97
        netstat command, 93 to 97
        packet loss, 91 to 92
        ping command, 90 to 92
        software checks, 90
        third-party diagnostic programs, 90
    UUCP over, 236 to 237
TCP/IP protocol suite, 11 to 26
    data communications, 19 to 24
        data encapsulation, 20 to 24
    described, 3
    displaying statistics, 94 to 95
    further information, 24 to 26
        books, 24
        FYIs, 25
        RFCs, 24 to 26
    OSI Reference Model, 12 to 13
    overview, 11 to 12
    standard services, 77
    TCP/IP protocol architecture model, 13 to 19
        application layer, 14, 16 to 19
        data-link layer, 14
        Internet layer, 14, 15
        physical network layer, 14
        transport layer, 14, 16
telephone lines
    PPP requirement, 127
    UUCP configuration, 186
telephone numbers in Systems file, 194
telnet program, 17

Telnet protocol, 17
temporary (TM) UUCP data files, 230
/tftboot directory creation, 74
tftp
  *See also* ftp
  network configuration server booting protocol, 48
  program described, 18
Th *Time* field entry, 192
three-way handshake, 22
*Time* field of Systems file, 191, 216
TLI dialer type, 202
TLI network, 202
TLIS dialer type, 202
TM UUCP temporary data files, 230
/tmp/.asppp.fifo file, 114
tokens (dialer token pairs), 202 to 205
TOO MANY LOCKS message, 243
TOO MANY SAVED C FILES message, 242
topology, 40 to 42
transfer speed for UUCP communication link, 193 to 194, 201
transferring files, *See* file transfers (UUCP)
Transmission Control Protocol, *See* TCP protocol
Transmission Control Protocol / Internet Protocol, *See* TCP / IP networks; TCP / IP protocol suite
transport layer
  data encapsulation, 21 to 22
  OSI, 13
  packet life cycle
    receiving host, 24
    sending host, 21 to 22
  TCP / IP
    described, 14, 16
    TCP protocol, 16
    UDP protocol, 16
Transport Level Interface Network (TLI), 202
trivial file transfer protocol, *See* tftp
troubleshooting
checking PPP links, 149 to 164
  connectivity, 151
  diagnostics, 155 to 164
  hardware, 150
  interface activity, 151
  interface status, 150 to 151
  local routing tables, 152 to 153
  order for checks, 149 to 150
  packet flow, 153 to 154
  packet reception, 151
  permissions, 153
PPP diagnostics, 155 to 164
  analyzing diagnostic output, 156 to 164
  communications between local and remote hosts, 160 to 164
  debug level, 155, 182
  described, 155
  editing asppp.cf file, 155
  host and modem setup, 156 to 159
  obtaining diagnostic information, 155
  setting diagnostics for your machine, 155
TCP / IP networks, 89 to 97
  displaying packet contents, 97
  general methods, 89 to 90
  ifconfig command, 92 to 93
  logging routing daemon actions, 97
  netstat command, 93 to 97
  packet loss, 91 to 92
  ping command, 90 to 92
  software checks, 90
  third-party diagnostic programs, 90
UUCP, 239 to 245
  ASSERT error messages, 241, 241 to 243
  checking basic information, 241
  checking error messages, 240 to 245
  checking Systems file, 239

commands for troubleshooting, 241
debugging transmissions, 239 to 240
faulty modem or ACU, 239
STATUS error messages, 241, 244 to 245
Tu *Time* field entry, 192
turning off
*See also* stopping
CLOCAL flag, 196
echo checking, 196, 209
RDISC, 87 to 88
RIP, 126 to 127
turning on
*See also* starting
CLOCAL flag, 196
echo checking, 196, 209
enabling dialback through chat script, 197
network configuration daemons, 74 to 75
space-saving mode, 87
*Type* field
Devices file, 199 to 200
Systems file, 193

## U

UDP protocol
described, 16
displaying statistics, 94 to 95
services in /etc/inet/services file, 71
UDP packet process, 22
uname -n command, 214
UNIX "r" commands, 18
UNIX-to-UNIX Copy Program, *See* UUCP
up option of ifconfig, 141
Usenet, 185
User Datagram Protocol, *See* UDP protocol
User keyword of *Permit-type* field, 227
*User-job-grade* field of Grades file, 225, 226
/usr/bin/cu program
checking modems or ACUs, 239
described, 188
multiple or different configuration files, 189, 213 to 214
printing Systems lists, 214
/usr/bin/uucp program
debugging transmissions, 240
described, 188
home directory of login ID, 187
permissions for forwarding operation, 224
uucico execution by, 187
/usr/bin/uulog program, 187, 241
/usr/bin/uupick program, 188, 239
/usr/bin/uustat program, 189, 239
/usr/bin/uuto program
described, 188
removing public directory files, 239
uucico execution by, 187
/usr/bin/uux program
described, 188
uucico execution by, 187
/usr/lib/uucp/uucheck program, 188, 241
/usr/lib/uucp/uucleanup program, 188
/usr/lib/uucp/Uutry program, 188, 240
/usr/sbin/aspppd PPP link manager
described, 112
FIFO file, 114
killing and restarting, 156
verifying if PPP running, 146
/usr/sbin/aspppls PPP login service
described, 113
FIFO file, 114
/usr/sbin/in.rdisc program
described, 82
dynamic routing selection and, 85
logging actions, 97
turning off RDISC, 87 to 88
/usr/sbin/in.routed daemon
described, 82

killing, 153
logging actions, 97
restarting, 153
space-saving mode, 82, 87
verifying if running, 153
/usr/sbin/inetd daemon
checking if running, 90
in.uucpd invoked by, 187
services started by, 77
/usr/sbin/ping command, 90 to 92
checking PPP connection, 151
described, 90
running, 91 to 92
syntax, 90 to 91
verifying if PPP running, 147
/usr/sbin/route command, 153
uucheck program, 188, 241
uucico daemon
adding UUCP logins, 233 to 234
described, 186 to 187
Dialcodes file and, 212
maximum simultaneous executions, 189, 228 to 229
multiple or different configuration files, 189, 191, 213 to 214
printing Systems lists, 214
Systems file and, 191
uusched daemon and, 187
Uutry program and, 188
uucleanup program, 188
UUCP
administrative files, 229 to 231
administrative programs, 187 to 188
callback option, 218 to 219
configuring
adding UUCP logins, 233 to 234
running UUCP over TCP/IP, 236 to 237
daemons
overview, 186 to 187
PPP diagnostics, 156 to 159
database files, 189 to 229
*See also specific database files*
basic configuration files, 190
described, 114, 189 to 190
multiple or different files, 189, 191, 213 to 214
PPP configuration, 136 to 138, 190
PPP diagnostics, 157 to 159
described, 185
directories
administration, 187
error messages, 241
public directory maintenance, 238
displaying log files, 187
file transfers
daemon, 186 to 187
permissions, 216 to 218
troubleshooting, 239 to 240
work files C., 230 to 231
forwarding operation, 224
hardware configurations, 186
log files
cleanup, 236
displaying, 187
“login shell”, 187
logins
adding, 233 to 234
privileges, 221 to 222
mail accumulation, 238
maintenance, 238 to 239
node name
alias, 189, 217
remote computer, 191, 214
overriding parameters manually, 224 to 225
passive mode, 216
polling remote computers, 189, 224
privileged logins and passwords, 221 to 222
public directory maintenance, 238
remote execution
commands, 216, 219 to 222
daemon, 187
work files C., 230 to 231
security
*See also* Permissions file

COMMANDS option of Permissions file, 219 to 220
setting up, 237
sticky bit for public directory files, 238
VALIDATE option of Permissions file, 221 to 222
shell scripts, 234 to 236
Solaris version, 185
spool
clean-up program, 188
job grade definitions, 225 to 228
scheduling daemon, 187
STREAMS configuration, 228
transfer speed, 193 to 194, 201
troubleshooting, 239 to 245
ACU faulty, 239
ASSERT error messages, 241, 241 to 243
checking basic information, 241
checking error messages, 240 to 245
checking Systems file, 239
commands for troubleshooting, 241
debugging transmissions, 239 to 240
modem faulty, 239
STATUS error messages, 241, 244 to 245
user programs, 188 to 189
uucp program
debugging transmissions, 240
described, 188
home directory of login ID, 187
permissions for forwarding operation, 224
uucico execution by, 187
uucppublic directory maintenance, 238
uudemon.admin shell script, 235
uudemon.cleanup shell script, 236
uudemon.crontab file, 234 to 235
uudemon.hour shell script
described, 235
uusched daemon execution by, 187
uuxqt daemon execution by, 187
uudemon.poll shell script, 224, 235
uudirect keyword of *DTP* field, 202
uulog program, 187, 241
uuname command, 241
uupick program
described, 188
removing public directory files, 239
uusched daemon
described, 187
maximum simultaneous executions, 189, 228 to 229
uudemon.hour shell script call, 235
uustat program
checking modems or ACUs, 239
described, 189
uudemon.admin shell script for, 235
uuto program
described, 188
removing public directory files, 239
uucico execution by, 187
Uutry program, 188, 240
uux program
described, 188
uucico execution by, 187
uuxqt daemon
described, 187
maximum simultaneous executions, 189, 228 to 229
uudemon.hour shell script call, 235


## V


-v option of uucheck program, 241
VALIDATE option of Permissions file, 221 to 222
COMMANDS option, 219 to 220
/var/adm/log/asppp.log file
described, 114
PPP diagnostics

communications between local and remote hosts, 160 to 164
host and modem setup, 156 to 159
obtaining diagnostic information, 155
/var/spool/uucppublic directory
maintenance, 238
/var/uucp/.Admin/errors directory, 241
/var/uucp/.Status directory, 241
version keyword, 182
virtual networks
configuring, 172 to 175
asppp.cf file, 174 to 175
/etc/passwd and /etc/shadow files, 174
hosts database, 173
IP address issues, 173
networks database, 174
described, 110 to 111
interface support, 103
network number assignment, 126
requirements, 123
sample network, 172

## W

WAN, *See* wide-area network (WAN)
We *Time* field entry, 192
wide-area network (WAN)
examples, 9
Internet
described, 9
domain name registration, 12
security information, 9
LAN access, 9
security issues, 9
Usenet, 185
wildcards in bootparams database, 67
will_do_authentication keyword
associated string, 176
defined, 178
syntax, 177
Wk *Time* field entry, 192
work (C.) UUCP files
cleanup, 236
described, 230 to 231
WRITE option of Permissions file, 217
NOWRITE option, 218
WRONG MACHINE NAME message, 245
WRONG ROLE message, 242
WRONG TIME TO CALL message, 244

## X

X. UUCP execute files
cleanup, 236
described, 231
uuxqt execution, 187
XMV ERROR message, 243

## Z

Copyright 1995 Sun Microsystems, Inc., 2550 Garcia Avenue, Mountain View, Californie 94043-1100 USA.

Tous droits réservés.Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie et la décompliation. Aucune partie de ce produit ou de sa documentation associée ne peuvent Être reproduits sous aucune forme, par quelque moyen que ce soit sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il en a.

Des parties de ce produit pourront etre derivees du système UNIX®, licencié par UNIX Systems Laboratories Inc., filiale entierement detenue par Novell, Inc. ainsi que par le système 4.3. de Berkeley, licencié par l'Université de Californie. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fourmisseurs de Sun.

LEGENDE RELATIVE AUX DROITS RESTREINTS : l'utilisation, la duplication ou la divulgation par l'administation americaine sont soumises aux restrictions visées a l'alinéa (c)(1)(ii) de la clause relative aux droits des données techniques et aux logiciels informatiques du DFAR 252.227- 7013 et FAR 52.227-19.

Le produit décrit dans ce manuel peut Être protege par un ou plusieurs brevet(s) americain(s), etranger(s) ou par des demandes en cours d'enregistrement.

MARQUES
Sun, Sun Microsystems, le logo Sun, SunSoft, le logo SunSoft, Solaris, SunOS, SunLink,OpenWindows, DeskSet, ONC, ONC+, et NFS sont des marques deposées ou enregistrées par Sun Microsystems, Inc. aux Etats-Unis et dans certains autres pays. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays, et exclusivement licenciée par X / Open Company Ltd. OPEN LOOK est une marque enregistrée de Novell, Inc., PostScript et Display PostScript sont des marques d'Adobe Systems, Inc.

Toutes les marques SPARC sont des marques deposées ou enregitrées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. SPARCcenter, SPARCcluster, SPARCompiler, SPARCdesign, SPARC811, SPARCengine, SPARCprinter, SPARCserver, SPARstation, SPARCstorage, SPARCworks, microSPARC, microSPARC II et UltraSPARC sont exclusivement licenciées a Sun Microsystems, Inc. Les produits portant les marques sont basés sur une architecture développée par Sun Microsytems, Inc.

Les utilisateurs d'interfaces graphiques OPEN LOOK® et Sun™ ont été développés par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnait les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique, cette licence couvrant aussi les licencies de Sun qui mettent en place OPEN LOOK GUIs et qui en outre se conforment aux licences écrites de Sun.

Le système X Window est un produit du X Consortium, Inc.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" SANS GARANTIE D'AUCUNE SORTE, NI EXPRESSE NI IMPLICITE, Y COMPRIS, ET SANS QUE CETTE LISTE NE SOIT LIMITATIVE, DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DES PRODUITS A REPONDRE A UNE UTILISATION PARTICULIERE OU LE FAIT QU'ILS NE SOIENT PAS CONTREFAISANTS DE PRODUITS DE TIERS.

CETTE PUBLICATION PEUT CONTENIR DES MENTIONS TECHNIQUES ERRONEES OU DES ERREURS TYPOGRAPHIQUES. DES CHANGEMENTS SONT PERIODIQUEMENT APPORTES AUX INFORMATIONS CONTENUES AUX PRESENTES, CES CHANGEMENTS SERONT INCORPORES AUX NOUVELLES EDITIONS DE LA PUBLICATION. SUN MICROSYSTEMS INC. PEUT REALISER DES AMELIORATIONS ET / OU DES CHANGEMENTS DANS LE(S) PRODUIT(S) ET / OU LE(S) PROGRAMME(S) DECRITS DANS DETTE PUBLICATION A TOUS MOMENTS.