

Configuring and Managing MPE/iX Internet Services

HP 3000 MPE/iX Computer Systems

Edition 2



Manufacturing Part Number: 32650-90891

E0300

U.S.A. March, 2000

Notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for direct, indirect, special, incidental or consequential damages in connection with the furnishing or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. 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. Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19 (c) (1,2).

Acknowledgments

UNIX is a registered trademark of The Open Group. Windows and Windows NT and registered trademarks of Microsoft Corporation.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304 U.S.A.

© Copyright 1997, 1998, 2000 by Hewlett-Packard Company

1. Introduction to Internet Services

Overview of Internet Services	15
Summary of HP 3000 Internet Services	16
System Requirements	16
Verifying Installation of Internet Services Files	17
Using Domain Name Resolver	18
Sample Configuration Files	18
Linking Configuration Files	18
Installed Configuration Files	19
Services File	20
Creating and Linking the Services File	20
Editing the Services File	21
Protocols File	23
Creating and Linking Protocols File	23
Viewing Protocols File	23
Other Sources of Information	24

2. Internet Daemon

Overview of inetd	25
Stream Services	25
Datagram Services	25
Internal Services Provided by inetd	26
inetd Files	26
inetd Configuration File	27
Creating and Linking inetd Configuration File	27
Adding New Services to inetd Configuration	27
inetd Security File	29
Creating and Linking inetd Security File	30
Updating inetd Security File	30
Starting and Stopping inetd	32
Starting inetd From a Job	32
Starting JINETD Automatically	33
Passwords on JINETD	33
Starting inetd Interactively	33
Error and Status Reporting for inetd	33
Stopping inetd	34
Summary of inetd Command Line Options	34
Using inetd Message Logging	35
Connection Logging	35
Enable and Disable Connection Logging	35
Troubleshooting inetd	36
Implementation Differences	38

3. Telnet Service

Overview of Telnet Service	40
--------------------------------------	----

Verifying Installation of Telnet Files	41
Configuring Telnet Server	42
Editing the Services File	42
Adding Telnet Service to inetd Configuration	42
Troubleshooting Telnet	44
Implementation Differences	46
4. BOOTP Service	
Overview of bootpd	48
bootpd Files	48
Configuring bootpd	48
Editing the Services File	48
Adding BOOTP Server to inetd Configuration	49
The bootpd Configuration File	50
Creating and Linking bootpd Configuration File	50
Editing the bootpd Configuration File	50
Adding Client and Relay Data to bootpd Configuration File	50
Syntax of bootpd Configuration Entries	51
Tags Used in bootpd Configuration File	52
Editing Tips	53
Sample bootpd Configuration Files	54
Starting bootpd	55
Starting bootpd Under inetd	55
Command Line Options for bootpd	55
Troubleshooting bootpd	56
Diagnostic Options	56
Sample Diagnostic Results	57
Implementation Differences	57
5. TFTP Service	
Overview of tftpd	60
Configuring tftpd	60
Editing the Services File	60
Adding TFTP Service to inetd Configuration	60
Starting tftpd	62
Troubleshooting tftpd	63
Implementation Differences	63
6. REMSH Service	
Overview of remsh Service	66
Verifying Installation of remsh Files	66
Configuring remsh Client	66

Editing the Services File	67
UNIX Configuration	67
Using remsh	68
MPE/iX Examples	68
POSIX Examples	69
Troubleshooting remsh	70
Implementation Differences	70

7. Samba/iX Services

Overview of Samba/iX	74
Introduction to Samba	74
Features of Samba/iX	74
Benefits of Using Samba/iX	75
Major Components of Samba/iX	75
SMB Protocol	77
Example of SMB Conversation	78
Samba/iX Configuration File Options	79
Global Configuration Options	80
Global Printer Service Options	82
Controlling User Access Rights	83
Share Configuration Options	84
Configuring the Shares for File Sharing	85
Configuring a Printer Section for Printer Sharing	86
Starting and Stopping Samba/iX	88
Starting Samba/iX	89
Disable Resource Sharing	89
Stopping Samba/iX	91
Initial Test With smbclient Utility	91
Initial Test From a PC Client at DOS Prompt	93
Samba/iX Share level Security Mode	94
Samba/iX Server Security Mode	95
Troubleshooting Samba/iX Server	96
Prerequisites	96
Troubleshooting Procedures	96
Using Logfiles of Samba/iX	99

8. DNS BIND/iX

Introduction	102
Explanation of Terms	103
Overview of DNS BIND/iX	104
DNS BIND/iX Component Files	105
Server Configuration File named.conf	105
Configuring Master Zones	107
Configuring Slave Zones	107
Data Files	108

Root Cache Data (Hint File)	108
Explaining DNS Database Files	108
How to Run The DNS Server	113
Configuring the DNS Resolver	114
List of Utilities	115
DNS and Electronic Mail	117
MX Records	117
DNS BIND Troubleshooting Steps	118

9. Apache for MPE/iX

Introduction	122
Feature Set	122
Major Components	124
Hardware Requirements	125
Steps for Starting Apache	125
Prepare Your HP 3000 for Network Access	126
Configure Apache	127
Edit your Configuration File	127
Edit the JHTTPD Job Stream File	127
Start the HTTPD Web Server Program	128
Verify that Apache is Running	128
Stopping Apache	129
Error Logging	130
Adding Documents	130
Troubleshooting	130
Unsupported Functionality	131
Performance	131
Additional Documentation	132

A. Samba/iX Sample Configuration File

B. BIND 8 Configuration File

C. BIND 8.1 Enhanced Features

BIND 8 Highlights	147
BIND Configuration File Guide — Logging Statement	147
BIND Configuration File Guide—Zone Statement	152
BIND Configuration File Guide — Options Statement	153
Converting From BIND 4.9.x	159

D. Server Configuration Migration

E. Configure and Run Syslog/iX

Figures

Figure 7-1. . HP 3000 Interoperating With Microsoft Platforms	75
Figure 7-2. . SMB Protocol	77
Figure 7-3. . SMB NegProt Connection	78
Figure 7-4. . SMB Sesssetup Connection	78
Figure 7-5. . SMB TconX Connection	79
Figure 7-6. . ADD a Printer	87
Figure 7-7. . Connect to the HP 3000 Shares.	88
Figure 7-8. . View the HP 3000 Share	88
Figure 7-9. . smbclient for MPE/iX (1)	92
Figure 7-10. . smbclient for MPE/iX (2)	93
Figure 7-11. . Display Available Shares From a PC Client.	94
Figure 8-1. . Labeling Nodes.	119

Tables

Table 1-1.. Summary of HP 3000 Internet Services	16
Table 1-2.. Configuration Files	19
Table 2-1.. The Internet Daemon Files.....	26
Table 4-1.. Files for bootpd	48
Table 7-1.. Major Components	76

Preface

This manual describes how to configure and operate Internet Services on the HP 3000. It is written for members of the system administration staff who have been assigned system manager (SM) or system supervisor (OP) capability and who are responsible for installing, configuring and managing system and network software. As such, it presumes a good understanding of networking concepts and familiarity with HP 3000 system operations. *Configuring and Managing Internet Services* is *not* intended for the end user of Internet Services such as `tftp` and `telnet`.

This manual is organized into the following chapters:

Chapter 1 , “Introduction to Internet Services,” describes in summary fashion each of the utilities that comprise the Internet Services product. It also includes instructions for installing and configuring the services file and protocols file.

Chapter 2 , “Internet Daemon,” describes the function and configuration of the Internet daemon `inetd` and provides troubleshooting guidelines.

Chapter 3 , “Telnet Service,” explains how to configure the `telnet` server and offers troubleshooting guidelines.

Chapter 4 , “BOOTP Service,” describes how to configure the Bootstrap Protocol daemon, provides examples that show how to add `bootp` clients and routing instructions to the HP 3000, and offers troubleshooting guidelines.

Chapter 5 , “TFTP Service,” describes how to configure the Trivial File Transfer Protocol daemon and explains `tftpd` security considerations and troubleshooting guidelines.

Chapter 6 , “REMSH Service,” describes how `remsh` or remote shell is used to connect to a specified host and execute a command on that host. This remote shell or `remsh` is available with version C.60.00 of the MPE/iX operating system.

Chapter 7 , “Samba/iX Services,” describes how the suite of programs work together to allow clients to access a server’s file space and printers, via the Server Message Block (SMB) file server.

Chapter 8 , “DNS BIND/iX,” describes BIND and its implementation of Domain Name System (DNS).

Appendix A , “Samba/iX Sample Configuration File,” shows an example of the `samp-smb.conf` configuration file.

Appendix B , “BIND 8 Configuration File,” describes the BIND 8 configuration file.

Appendix C , “BIND 8.1 Enhanced Features,” describes the options and enhanced features available.

Appendix D , “Server Configuration Migration,” describes configuration migration utilities.

Appendix E , “Configure and Run Syslog/iX,” describes the parameters in a syslog configuration file.

Glossary

1 Introduction to Internet Services

The HP 3000 Internet Services consist of a set of programs that help the HP 3000 computer exchange information with other nodes on the **internet**. The Internet Services offered on the HP 3000 are a subset of the Internet Services available on the HP 9000, which were previously called the ARPA Services. This introductory chapter:

- Provides an overview of the Internet Services
- Lists the system requirements for using Internet Services
- Shows how to verify the installation of the set of configuration and program files for Internet Services that were delivered with the MPE/iX Fundamental Operating Software.
- Lists configuration files
- Describes two configuration files that all of the Internet Services use, the **protocols** file and the **services** file, and how to install and edit them.

At the end of this introductory chapter, there is a list of additional manuals that may be helpful.

By and large, the subset of Internet Services running on an MPE/iX system are identical to those available on UNIX machines. There are, however, some differences between them. If you are an experienced HP-UX system administrator and you plan to skim the information in this manual, pay attention to the “Implementation differences” sections at the end of each chapter. They describe the important differences between MPE/iX version of Internet Services and the HP-UX version of Internet Services.

Overview of Internet Services

Internet Services on the HP 3000 consist of eight individual services that enable the HP 3000 to communicate with other nodes on an internetwork. The program and configuration files needed to run Internet Services is part of the MPE/iX Fundamental Operating Software. No separate software product is necessary to use Internet Services.

The services are briefly described in Table 1-1. on page 16.

Summary of HP 3000 Internet Services

Table 1-1. Summary of HP 3000 Internet Services

Service	Description
inetd	The Internet daemon <code>inetd</code> is the master server for the group of Internet Services rather than an individual network service. You must install and configure <code>inetd</code> on your system to use the other services as listed below.
telnet	The telnet server uses the standard virtual terminal protocol to allow users on a remote node that supports Internet Services to log on and run most applications on the host HP 3000.
bootpd	The Bootstrap Protocol daemon, or <code>bootpd</code> , is used to boot, or start, devices such as routers, printers, X-terminals and diskless workstations. Client systems use <code>bootpd</code> to find their own IP address and the name of the boot file to load into memory and execute.
tftpd	The Trivial File Transfer Protocol daemon <code>tftpd</code> is used to transfer the boot files needed to start network devices. In this implementation of Internet Services, <code>tftpd</code> enables an HP 3000 to boot network printers.
remsh	The remote shell client allows a user on an HP 3000 to access a remote UNIX host and execute a UNIX command or script without logging on.
ftp	The File Transfer Protocol (FTP) is an ARPA service that allows users to transfer files among other networked systems. FTP is the file transfer program that uses the ARPA standard File Transfer Protocol. FTP can be used with systems supporting the ARPA FTP service such as other HP systems, UNIX systems, and non-UNIX systems.
Samba	Samba/iX is a suite of programs which allow clients to access a server's file space and printers via the Server Message Block (SMB) protocol. It allows the MPE/iX shell operating system to act as a file and printer server for SMB clients, which are primarily, Windows NT, Windows 95 and Windows for Workgroups
DNS	BIND (Berkeley Internet Name Domain) is an implementation of the Domain Name System (DNS). A complete implementation of DNS BIND/iX is available on MPE/iX. DNS BIND/iX will enable MPE/iX host to act as a DNS server, both responding to queries as well as communicating with other DNS servers on the local network and the Internet.

NOTE Throughout this manual, the term `daemon`, which is familiar to UNIX users, and the term `server` are used interchangeably.

System Requirements

The Internet Services program and configuration files come with version C.55.00 or greater, of the MPE/iX Fundamental Operating Software (FOS). (The exception to this is the Telnet Client, which was made available to customers on the earlier version of MPE/iX, C.50.00.) As part of MPE/iX FOS, Internet Services can run on any Precision Architecture-RISC model of the HP 3000. They are not available on earlier "classic"

HP 3000 computers running MPE V.

To run Internet Services, you must:

- Configure one or more network interface link cards that support **TCP/UDP/IP** communications protocol.
- Configure the Net Transport communications software which uses the **TCP/UDP/IP** protocol.

The necessary software and at least one TCP/UDP/IP network interface card is delivered with each PA-RISC HP 3000 system. Internet Services runs on top of the Net Transport software and therefore runs over any type of link supported by Net Transport.

Verifying Installation of Internet Services Files

If you have installed or updated to version C.60.00 of MPE/iX, use the following steps to verify that the Internet Services files exist on your system:

1. If necessary, log on the system as `MANAGER.SYS`.
2. Enter a `LISTFILE` command for the `NET` group of the `SYS` account:
:`LISTFILE @.NET.SYS`
3. Check the list displayed on your screen and make sure that you have the following files:

```
BOOTPD          BOOTPQRY
BPTABSMP        INETD
INSECSMP        INSVXL
JINETD          PROTSAMP
REMSH           SERVSAMP
TFTPD
```

4. Run the Node Manager Maintenance utility to verify that you have successfully installed the set of Internet Services files (except for Telnet files, which you will check in Chapter 3, "Telnet Service.")

```
:NMMMAINT,73
```

You will see information similar to the following:

```
: nmmmaint,73
NMS Maintenance Utility 32098-20014 B.00.09 (C) Hewlett Packard Co. 1984

WED, JUL 23, 1997, 11:08 AM Data comm products build version: N.55.15

Subsystem version ID's:
```

```
Internet Services for the HP3000 module versions:
```

```
NM program file: INETD.NET.SYS           Version: B0001003
NM program file: BOOTPD.NET.SYS          Version: B0001003
NM program file: BOOTPQRY.NET.SYS        Version: B0001002
NM program file: TFTPD.NET.SYS           Version: B0001002
```

```

NM program file: REMSH.NET.SYS           Version:  B0001003
XL procedure:    INSVXL_SECURE_VERS      Version:  B0001004
XL procedure:    INSVXL_IPCSEC_VERS      Version:  B0001002
XL procedure:    INSVXL_NSRW_VERS        Version:  B0001003
XL procedure:    INSVXL_NETOF_VERS       Version:  B0001002
XL procedure:    INSVXL_SYSLOG_VERS      Version:  B0001003
XL procedure:    INSVXL_SIGNAL_VERS      Version:  B0001002
XL procedure:    INSVXL_GETTIME_VERS     Version:  B0001003

```

Internet Services for the HP3000 overall version = B.00.01

The final line of information, which displays the current overall version of these software files, is useful when you need to call the Hewlett-Packard support staff.

5. Check for any error messages, such as a module is missing, or a message telling you of a version mismatch, for example:

```

Version levels differ in one or more modules.  (NMERR 103)
Internet Services for the HP3000 overall version =  ?..??.??

```

Using Domain Name Resolver

To use the domain name resolver to resolve domain names to their IP addresses, you will need to configure a set of ASCII files on each node that contain the necessary information. Refer to the “Configuring the DNS Resolver” section of this chapter, or the *HP 3000/iX Network Planning and Configuration Guide*.

Sample Configuration Files

When you install or update to version C.60.00 of MPE/iX, a set of *sample* configuration files is automatically copied to the NET group of the SYS account for you. For example, INCNFSMP is the name of the sample `inetd` configuration file. These files were named and installed in this form to prevent overwriting any genuine configuration files already in use.

To view the group of files installed in NET.SYS, enter:

```
:LISTFILE @.NET.SYS
```

To configure Internet Services, you will do one of two things:

- If there are configuration files already in use, you will add the information needed to use each of the Internet Services to those files.
- If you are not already using Internet Services configuration files, you will use the sample configuration files that were installed with the FOS as templates for your own set. In this case, you will use the `COPY` command to create each of the configuration files, then create a symbolic link from a file name in the POSIX name space to the actual file, which exists in the MPE name space. (Linking the files is explained next.) Finally, you will edit the new configuration files to suit your needs.

Linking Configuration Files

The Internet Services software looks for some of its configuration files in the POSIX name space and not in the MPE name space. For example, it accesses the `/etc` directory and looks for the file named `inetd.conf` to read `inetd` configuration data. It does not look for

the file `INETDCNF.NET.SYS`.

Rather than create two copies of the configuration file, one for each name space, Hewlett-Packard recommends that you create a symbolic link from a POSIX-named file to the MPE-named file. The instructions in the remainder of this manual describe this process. Linking the files, as opposed to making another copy of each one, offers three important advantages.

- Linking the file ensures consistency of content because regardless of which name you use to access the file, you will be reading or updating the same file.
- Giving the file a name in each name space allows you to view the file from either the POSIX or the MPE name space, but it is recommended that you use an MPE text editor to make changes. This is due to potential conflicts with the MPE/iX EOF marker if any lines are added using a POSIX editor program.
- Making the POSIX name point to the MPE name ensures that the file will be backed up with standard MPE `STORE` procedures in case you haven't modified your `STORE` command to back up new or changed files in the POSIX name space.

Installed Configuration Files

If you install and configure all of the Internet Services according to the instructions in this manual, you will have the set of files described in Table 1-2.

Table 1-2. Configuration Files

Sample name	MPE name space	HFS name space	Description
<code>SERVSAMP.NET.SYS</code>	<code>SERVICES.NET.SYS</code>	<code>/etc/services</code>	The services name file, which associates an official service name and alias with the port number and protocol that a service uses. You will edit the services file for each new service you are adding to your system. The executing program uses the file named <code>SERVICES.NET.SYS</code> .
<code>PROTSAMP.NET.SYS</code>	<code>PROTOCOL.NET.SYS</code>	<code>/etc/protocols</code>	The file containing a list of protocols known to the system and the identification number and one or more aliases for each. You will rarely, if ever, need to edit this file. The executing program uses the file named <code>PROTOCOL.NET.SYS</code> .
<code>INCNFSMTP.NET.SYS</code>	<code>INETDCNF.NET.SYS</code>	<code>/etc/inetd.conf</code>	The configuration file for the Internet daemon <code>inetd</code> , which determines which installed Internet Services are available to users. The executing program uses the file named <code>/etc/inetd.conf</code> .

Table 1-2. Configuration Files

Sample name	MPE name space	HFS name space	Description
INSECSMP. NET.SYS	INETDSEC.NET.SYS	/usr/adm/inetd .sec	The optional security file for inetd, which lets you control access to individual services by specific nodes. The executing program uses the file named /usr/adm/inetd.sec.
BPTABSMP. NET.SYS	BOOTPTAB.NET.SYS	/etc/bootptab	The configuration file for the Bootstrap protocol daemon, bootpd. The executing program uses the file named /etc/bootptab.

For each individual service you install, you will always edit the **services** file and the `inetd` configuration file. It is unlikely that you will need to edit the protocols file. The remainder of this chapter explains the **services** and **protocols** file. Chapter 2, “Internet Daemon,” explains working with the `inetd` configuration files.

Services File

The services file associates an official service name and alias with the port number and protocol that a service uses. You will edit the services file for each new service that you want to add to your system. The remaining chapters in this book, which describe the configuration of individual services, will assume that you know the following information. And, of course, you can refer back to this section as needed.

Creating and Linking the Services File

You may already have a services file installed on your system. If you know that you have such a file, and it is accessible by the POSIX file name `/etc/services` you may skip these steps.

If you do not have a services file, follow these steps to create the file and link to it. If you have such a file, but are unsure whether or not it is linked, perform step 2 only.

1. Create your own services file by using the `COPY` command to rename the sample file. Enter:

```
:COPY SERVSAMP.NET.SYS, SERVICES.NET.SYS
```

2. Create a symbolic link from a file named `/etc/services` in the POSIX name space to `SERVICES.NET.SYS`. Enter:

```
:NEWLINK /etc/services, SERVICES.NET.SYS
```

Editing the Services File

Use an MPE text editor to edit the file.

1. Open the services file with an MPE text editor.

The contents will resemble the following:

```
# This file contains the information about the services provided.
# Copy this file to SERVICES.NET.SYS if that file does not already exist.
#
# The form for each entry is:
# <official service name>    <port number/protocol name>    <aliases>
#
# See the Configuring and Managing MPE/iX Internet Services Manual
# for more information (HP Part No. 32650-90835).
#
# Note: The entries cannot be preceded by a blank space.
#
echo            7/tcp                # Echo
echo            7/udp                #
discard        9/tcp    sink null    # Discard
discard        9/udp    sink null    #
daytime        13/tcp                # Daytime
daytime        13/udp                #
chargen        19/tcp    ttytst source # Character Generator
chargen        19/udp    ttytst source #
ftp            21/tcp                #
telnet         23/tcp                #
time           37/tcp    timeserver  # Time
time           37/udp    timeserver  #
domain         53/tcp    nameserver  # Domain Name Service
domain         53/udp    nameserver  #
bootps         67/udp                # Bootstrap Protocol Server
bootpc         68/udp                # Bootstrap Protocol Client
tftp           69/udp                # Trivial File Transfer Protocol
DAServer       987/tcp               # SQL distributed access
shell          514/tcp    cmd         # Remote command no password used
```

2. For the service that you are installing, check the file to see if it has the appropriate entry. (Each chapter in the remainder of this manual has this information.) If not, enter the line in the file using the “Editing Tips” section, next, as a guideline.

NOTE For more information on FTP, refer to *Installing and Managing HP ARPA File Transfer Protocol Network Manager’s Guide* or *HP ARPA File Transfer Protocol User’s Guide*.

3. Save the file and exit the editor program.

Editing Tips

When you are editing the services file, use the following information to enter the information correctly.

- If you find the line that describes the service you are configuring, but it has been “commented out” (that is, preceded by a pound sign, #), the service has not yet been

enabled. To enable it, simply delete the pound sign *and* any spaces that precede the service name.

- If you need to type the line into the file:
 - use only lower case characters
 - enter the service name in the first column without any leading spaces
 - separate the individual fields on the line with any number of blanks or tab characters to improve readability

Protocols File

The protocols file contains a list of **protocols** known to the system, plus the identification number and one or more aliases for each. It is unlikely that you will need to edit the protocols file, but you may need to install and link it.

Creating and Linking Protocols File

You may already have a protocols file installed on your system. If you know that you have such a file, and it is accessible by the POSIX file name `/etc/protocols` you may skip these steps.

If not, follow the steps below to create and link the protocols file, `PROTOCOL.NET.SYS`. If you have such a file, but are unsure whether or not it is linked, perform step 2 only.

1. Use the `COPY` command to create the protocols file. Enter:

```
:COPY PROTSAMP.NET.SYS, PROTOCOL.NET.SYS
```

Make sure that you enter the singular form of protocol in the new MPE file name. That is, "PROTOCOL" and not "PROTOCOLS" should appear on the right side of the `COPY` command.

2. Create a symbolic link from `/etc/protocols` in the POSIX name space to `PROTOCOL.NET.SYS`. Enter:

```
:NEWLINK /etc/protocols, PROTOCOL.NET.SYS
```

Again, make sure that you enter the singular form of protocol in the new MPE file name `PROTOCOL.NET.SYS`.

Viewing Protocols File

Use an MPE text editor to open the file. It is unlikely that you will need to edit the file, but you can look at it now to familiarize yourself with its contents.

```
# This file associates protocol numbers with official protocol names and
# aliases. This allows the user to refer to a protocol by a symbolic
# name instead of a number. For each protocol a single line should be
# present with the following information:
#
# The form for each entry is:
# <official protocol name>    <protocol number>    <aliases>
#
# See the Configuring and Managing MPE/iX Internet Services Manual
# for more information (HP Part No. 32650-90835).
#
# Note: The entries cannot be preceded by a blank space.
#
11      tcp      6          TCP      # transmission control protocol
12      udp      17         UDP      # user datagram protocol
```

Other Sources of Information

You may find the following books useful when you are working with Internet Services:

- *Unix Network Programming* written by W. Richard Stevens. New Jersey: Prentice Hall, 1990
- *Telnet/iX User's Guide*

2 Internet Daemon

The **Internet daemon** `inetd` is the master server (sometimes called a “superserver”) for the Internet Services. When it is running, `inetd` listens for connection requests for the services listed in its configuration file and, in response to such requests, starts the appropriate server. You, as system manager, determine which Internet Services are available to your users by editing the `inetd` configuration file.

This chapter explains:

- How `inetd` behaves with **stream services** and with **datagram services**.
- How to edit the `inetd` configuration file so that it listens for connection requests from the specific Internet Services you want to use on your system.
- How to edit the optional security file for `inetd` which lets you control access to the Internet Services.
- How to use `inetd` logging capabilities to monitor and troubleshoot Internet Services.
- How to start and stop `inetd`.
- How to troubleshoot common problems that can occur with `inetd`.
- The implementation differences between `inetd` for MPE/iX and HP-UX.

Overview of inetd

The Internet daemon, or `inetd`, is the master server that coordinates the use of individual network services on your system. It listens for connection requests from other **nodes** on the network who want access to a service such as `tftpd` or `bootpd`. The Internet daemon checks if the requesting node has permission to use the service, starts the appropriate server if it does and, optionally, records information about the connection request.

Stream Services

The Internet daemon starts servers for both **stream services** and **datagram services**. For stream services, which use the **TCP/IP** protocol, `inetd` listens for connection requests on **stream sockets**. When it detects such a request, `inetd` determines which service the **socket** corresponds to and invokes a server for it. The server then handles incoming data, providing a reliable, full-duplex bytestream service to the requesting node. Once `inetd` has invoked the server, it returns to listening for other connection requests.

Datagram Services

For datagram services, which use the **UDP/IP** protocol, `inetd` listens for requests on **datagram sockets**. You can think of a datagram as a connection request and the message

all in one package. Unlike the TCP/IP protocol, UDP/IP does not provide any message acknowledgment, flow control or sequencing. It is the simplest possible service with the advantage of low communications overhead. When `inetd` detects an incoming datagram, it invokes a server for that message. Once a datagram has been delivered, the socket becomes available for another incoming datagram. That is, there is no “connection,” simply the delivery and receipt of the datagram. For this reason, datagram service is sometimes referred to as “connectionless” communication.

Internal Services Provided by inetd

The Internet daemon provides several internal trivial services which are described here.

Service	Description
<code>echo</code>	Returns a character to the socket that sent it
<code>discard</code>	Discards all input from socket
<code>chargen</code>	Generates characters and sends them to a socket
<code>daytime</code>	Returns the current time in a format readable by people.
<code>time</code>	Returns current time in a format useful to machines, for example, the number of seconds since Jan 1, 1970.

inetd Files

There are four files of importance as shown in Table 2-1., for configuring and using `inetd`. Once you have installed or updated to version C.60.00 or later, of MPE/iX, these files are located in the `NET` group of the `SYS` account.

Table 2-1. The Internet Daemon Files

File	Description
<code>INETD.NET.SYS</code>	The program file for <code>inetd</code> which is linked to the POSIX file <code>/etc/inetd</code> .
<code>INCNFSMP.NET.SYS</code>	The sample configuration file for <code>inetd</code> . You will copy the sample file to <code>INETDCNF.NET.SYS</code> , create a symbolic link from the POSIX file <code>/etc/inetd.conf</code> to <code>INETDCNF.NET.SYS</code> , and edit it as necessary.
<code>INSECSMP.NET.SYS</code>	The sample security file for <code>inetd</code> . You will copy this file to <code>INETDSEC.NET.SYS</code> , create a symbolic link from the POSIX file <code>/usr/adm/inetd.sec</code> to <code>INETDSEC.NET.SYS</code> , and edit it as necessary.
<code>JINETD.NET.SYS</code>	The job file that you will stream to start <code>inetd</code> and abort to stop <code>inetd</code> . You won't need to copy, link, or edit this file.

The remainder of this chapter explains how to copy, link and edit these files to create a working version of the Internet daemon on your system.

inetd Configuration File

The Internet daemon accesses the configuration data it needs by reading the file `/etc/inetd.conf` in the POSIX name space. When you install or update to version C.60.00 of MPE/iX, you receive a sample configuration file that you can use as a template for your own `inetd` configuration file if you don't already have one. This process involves two steps: creating the actual file in the MPE name space and creating a symbolic link that points from the POSIX file `/etc/inetd.conf` to the MPE file. The steps to create and link the file is explained later in this section. The reasons Hewlett-Packard recommends symbolic linking is explained in Chapter 1, "Introduction to Internet Services."

The Internet daemon reads its configuration file on three occasions:

- When `inetd` is started during normal system startup
- When `inetd` is started following a network shutdown as opposed to a system shutdown
- When you instruct an executing `inetd` to reread the configuration file after you have made changes to it that you wish to put into effect

Creating and Linking inetd Configuration File

You may already have a configuration file for `inetd` installed on your system. If you know that you have such a file, and it is accessible by the POSIX file name `/etc/inetd.conf` you may skip these steps.

If not, follow these steps to create the file and link to it. If you have such a file, but are unsure whether or not it is linked, perform step 2 only.

1. Create your own configuration file by using the `COPY` command to rename the sample file. Enter:

```
:COPY INCNFSMP.NET.SYS TO INETDCNF.NET.SYS
```

2. Create a symbolic link from `/etc/inetd.conf` in the POSIX name space to `INETDCNF.NET.SYS`. Enter:

```
:NEWLINK /etc/inetd.conf, INETDCNF.NET.SYS
```

3. Check the security provisions of the file and change them, if necessary. Hewlett-Packard recommends that only `MANAGER.SYS` has write access to `INETDCNF.NET.SYS`, and write and purge access to `/etc/inetd.conf`.

Adding New Services to inetd Configuration

There are two steps required to add a new service to the suite of Internet Services offered on your system. First you enter a line of information for the specific service to the `inetd` configuration file. Then you have `inetd` reread its configuration file, which is sometimes called reconfiguring the Internet daemon. In the unlikely event that `inetd` is not running when you edit the configuration file, you will invoke the new configuration by starting `inetd`. Starting `inetd` is explained later in this chapter.

To edit the `inetd` configuration file, do the following:

1. Open the configuration file with an MPE text editor.

The contents will resemble the following:

```
#####  
#  
# sample inetd configuration file  
#  
# For information on how to configure this file refer to the Configuring  
# and Managing Internet Services manual  
#  
# Note: The entries cannot be preceded by a blank space. Blank lines  
# and lines beginning with a pound sign(#) are ignored.  
#  
#####  
#  
# Internet server configuration database  
#  
echo          stream tcp nowait MANAGER.SYS internal  
echo          dgram  udp  nowait MANAGER.SYS internal  
daytime      stream tcp nowait MANAGER.SYS internal  
daytime      dgram  udp  nowait MANAGER.SYS internal  
time         stream tcp nowait MANAGER.SYS internal  
time         dgram  udp  nowait MANAGER.SYS internal  
discard      stream tcp nowait MANAGER.SYS internal  
discard      dgram  udp  nowait MANAGER.SYS internal  
chargen      stream tcp nowait MANAGER.SYS internal  
chargen      dgram  udp  nowait MANAGER.SYS internal  
#telnet      stream tcp nowait MANAGER.SYS internal  
#bootps      dgram  udp  wait   MANAGER.SYS /SYS/NET/BOOTPD bootpd  
#tftp        dgram  udp  wait   USER.TFTP  /SYS/NET/TFTPD tftpd  
#
```

2. Each of the services that run under inetd must have an entry in the configuration file. For example, the entry for the tftp program in INETDCNF.NET.SYS looks like this:

```
tftp dgram udp wait USER.TFTP /SYS/NET/TFTPTD tftpd
```

For the service that you are installing, check the file to see if it has the correct entry. (Each chapter in the remainder of this manual has this information. The meaning of the individual fields in an entry are explained later in this chapter.) If not, enter the line now using the “Editing Tips” section, as a guideline.

NOTE For more information on FTP, refer to *Installing and Managing HP ARPA File Transfer Protocol Network Manager's Guide* or *HP ARPA File Transfer Protocol User's Guide*.

3. Save the file and exit the editor program.

4. Signal inetd to reread the configuration file by entering the following command at the CI prompt:

```
INETD.NET.SYS -c
```

Or you may enter this command from the POSIX shell:

```
$/etc/inetd -c
```

Editing Tips

When you are editing the `inetd` configuration file, keep in mind these points:

- If you find the line, but it has been “commented out” (that is, preceded by a pound sign, #), the service has not yet been enabled. To enable it, simply delete the pound sign *and* any spaces that precede the service name.
- If you need to type the line into the file:
 - Use only lowercase characters
 - Enter the service name in the first column without any leading spaces
 - Separate the individual fields on the line with any number of blanks or tab characters to improve readability

Fields in an inetd Configuration File Entry

Each entry in the `inetd` configuration file conforms to a common format in which each of the fields has a specific purpose. For example, the entry for TFTP looks like this:

```
tftp dgram udp wait USER.TFTP /SYS/NET/TFTPD tftpd
```

Reading an entry from left to right, these fields are:

Field	Purpose
service name	The name of the service in the services file.
socket type	Either <code>stream</code> if the socket is a stream socket, or <code>dgram</code> if the socket is a datagram socket.
protocol	A valid protocol name, either <code>tcp</code> or <code>udp</code> , as entered in the protocols file.
wait state	One of two states, <code>wait</code> or <code>nowait</code> , that applies only to datagram sockets. The <code>wait</code> entry instructs <code>inetd</code> to execute only one datagram server for the specified socket at any one time. This is a single-threaded datagram server. The <code>nowait</code> entry instructs <code>inetd</code> to execute a datagram server for a specified socket whenever a datagram arrives, which frees the socket so that <code>inetd</code> can receive further datagrams. This is a multi-threaded datagram server.
user	The identification of the user when the server is running.
server program	The absolute path of the program that <code>inetd</code> executes when it receives a connection request.
arguments	Arguments to the server program, beginning with argument zero, which is the name of the program.

inetd Security File

There is an optional security file associated with `inetd` that allows you to control which nodes have access to the Internet Services available on your system. The `inetd` security

file will prevent `inetd` from starting a service unless the node making the request has permission to do so. Individual entries in the `inetd` security file determine which nodes are allowed or disallowed for a particular service.

The `inetd` security file is not the only security provided for Internet Services. It constitutes an extra layer of security in addition to the normal checks done by the services themselves. If the `inetd` security file does not exist, if a remote service is not listed in the security file, or if it is listed but it is not followed by the `allow` or `deny` key word, all remote hosts can attempt to use it. Such an attempt will succeed if it passes the security checks imposed by the requested service.

If `inetd` refuses a connection for security reasons, and `inetd` connection logging is enabled, a message is sent to the console indicating that there was an unsuccessful connection attempt.

Creating and Linking `inetd` Security File

You may already have a security file for `inetd` installed on your system. If you know that you have such a file, and it is accessible by the POSIX file name `/usr/adm/inetd.sec` you may skip these steps.

If not, follow the steps below to create the file and link to it. If you have such a file, but are unsure whether or not it is linked, perform step 2 only.

1. Create your own `inetd` security file by using the `COPY` command to rename the sample file. Enter:

```
:COPY INSECSMP.NET.SYS TO INETDSEC.NET.SYS
```

2. Create a symbolic link from `/usr/adm/inetd.sec` in the POSIX name space to `INETDSEC.NET.SYS`. Enter:

```
:NEWLINK /usr/adm/inetd.sec, INETDSEC.NET.SYS
```

3. Check the security provisions of the file and change them, if necessary. Hewlett-Packard recommends that only `MANAGER.SYS` has write access to `INETDSEC.NET.SYS`, and write and purge access to `/usr/adm/inetd.sec`.

Updating `inetd` Security File

Each line in the `inetd` security file contains a service name, a permission field, and the **IP addresses** or **domain names** of the hosts and networks allowed to use that service on your host system. You can open the file to view the current security restraints or to change them. To do so:

1. Open the security file with an MPE text editor. The contents will resemble the following:

```
# The lines in the file contain a service name, permission field and
# the Internet addresses or names of the hosts and/or networks
# allowed to use that service in the local machine.
# The form for each entry in this file is:
#
# <service name> <allow/deny> <host/network addresses, host/network names>
#
```

```
# For example:
#
# telnet          allow  10.3-5 192.34.56.5 ahost anetwork
#
# The above entry allows the following hosts to attempt to access your
system
# using telnet:
#
#             hosts in subnets 3 through 5 in network 10,
#             the host with Internet Address of 192.34.56.5,
#             the host by the name of "ahost",
#             all the hosts in the network "anetwork"
#
# tftp           deny   192.23.4.3
#
# The tftp entry denies host 192.23.4.3 to access your system using tftp
#
# Hosts and network names must be official names, not aliases.
# See the Configuring and Installing Internet Services Manual for more
# information.
```

The word `allow` or `deny` in the second column determines whether the list of remote hosts in the next field to the right has access to the specified service. If there is more than one line for a service, regardless of whether a statement indicates `allow` or `deny`, the `inetd` server ignores all but the last line.

2. Make any necessary editing changes. Refer to the following three sections, “Editing Tips”, “Using Wildcard Characters” and “Using Range Character” for more information.
3. Save your file and exit the editor.

Editing Tips

When you edit the `inetd` security file, remember the following points:

- To “comment out” a line, begin column 1 with a pound symbol (#). To enable a security provision that has been commented out, delete the pound symbol *and* any blank spaces preceding the service name.
- Enter the real service name, not the alias, of a valid service in the `inetd` configuration file.
- Separate the IP addresses and domain names by a white space. You may enter any mix of addresses and names. For example, the following entry denies Telnet access to host `hp22.cup.hp.com`, any hosts on the network named “testlan,” and the host with IP address `192.54.24.5`:

```
telnet deny hp22.cup.hp.com testlan 192.54.24.5
```
- To continue an entry on the next line, place a slash (/) *at the end of the line* to be continued. The Internet daemon will ignore a slash that appears in the middle of the line, continue reading to the end, and ignore the next line. In this case, it will probably misinterpret the entry and you will see an error message.

Using Wildcard Characters

You may use wildcard characters (*) in any of the fields of the address to specify permissions for a group of hosts or networks. This makes it more convenient to specify an

entire network, since you will not need to specify each host in that network. The following sample entry, for example, allows all hosts with network addresses starting with a 10, as well as the single host whose address is 192.54.24.5 to use Telnet:

```
telnet allow 10.* 192.54.24.5
```

You cannot use the wildcard character in combination with other integers in one part of an address field. For example, this entry in the `inetd` security file will generate an error message because the second field includes a 5 followed by the `*` character:

```
tftp deny 10.5*
```

Either integers *or* the wildcard character is allowed in one part of an address field.

Using Range Character

You may use the range indicator (`-`) in any of the fields of the address to specify which hosts or networks in a group are exempted from the permission assignment. This makes it more convenient to allow or deny a service for a subnet within the network you specify. The following sample entry, for example, denies hosts in subnets 3 through 5 of network 10 access to Telnet. Note that the wildcard character `*` at the end of the address lets you avoid specifying the individual hosts within the subnet.

```
telnetd deny 10.3-5.*
```

Starting and Stopping inetd

On the HP 3000, the instructions for starting the Internet daemon are contained in the job file `JINETD.NET.SYS`. When you stream `JINETD`, it invokes the daemon and reads the `inetd` configuration file to determine what services have been configured, and listens for connection requests for those services. Any messages relating to `inetd` are sent either to the console or to `$STDLIST` for `JINETD`, which is a spool file. The Internet daemon will continue to run, responding to requests for any of the configured services, until you stop it. The Internet daemon only terminates in an error state if there are no valid services listed in the configuration file.

Starting inetd From a Job

To start `inetd`, you stream the `JINETD` job. You may do this manually, by entering the `STREAM` command when the system is running, or you may include the `STREAM JINETD` command in the `SYSSTART` file to have `inetd` automatically started at system startup.

To start `inetd` manually:

1. Log onto your system as `MANAGER.SYS,NET`.
2. Check to make sure that `inetd` has not already been started by entering at the CI prompt:

```
:SHOWJOB JOB=@J
```


Look for the job logged on as `JINETD.NET.SYS` and, if it is not listed, continue with the next step.

3. At the CI prompt, enter `STREAM JINETD.NET.SYS`.

If you attempt to start `inetd` when it is already running, you'll see the following error message and the job will not be started:

```
An inetd is already running.
```

Starting JINETD Automatically

If you want to have the Internet daemon started automatically when your system starts up, add the `STREAM JINETD` command to the `SYSSTART` file. When you do, be sure that the stream command follows the network startup command `NETCONTROL START`.

Passwords on JINETD

When you stream the job file `JINETD.NET.SYS`, it logs on as `MANAGER.SYS`. As part of the installation of `inetd`, you must take care of any password requirements for this job. Two of the ways that you can do this include:

- Add the `MANAGER.SYS` passwords directly to the job file, then alter the file security afterwards so that only `MANAGER.SYS` can read it. For example:

```
:ALTSEC JINETD.NET.SYS; (R,W:CR;X,L:AC)
```
- Use the `PASSEXEMPT` parameter of the `JOBSECURITY` command (version C.60.00 and later) to control password exemption.

Starting inetd Interactively

You may also start `inetd` interactively, though this is not recommended for normal use. To do so, enter the following command at the CI prompt:

```
:INETD.NET.SYS
```

Or, from the POSIX shell enter this command:

```
$/etc/inetd
```

When you start `inetd` interactively, `$STDLIST` for the Internet daemon is your terminal. This means that all error and warning messages that normally go to `JINETD`'s spool file will appear on the screen.

Error and Status Reporting for inetd

While `inetd` is running, any errors and other status messages that it generates are recorded so that you can monitor its condition. All errors, regardless of their degree of seriousness, are sent to the `$STDLIST` device assigned to `inetd`. For example, if you streamed `JINETD`, error messages will appear in the spool file associated with that job. More critical errors are displayed on the system console in addition to being sent to `$STDLIST`. For more information, read "Using `inetd` Message Logging" later in this chapter.

Stopping inetd

To stop inetd, you abort the JINETD job. Stopping the inetd server (aborting JINETD) will cause subsequent incoming connection requests to be refused.

1. First find the number assigned to JINETD by entering:

```
:SHOWJOB JOB=@J
```

You will see a display of job information similar to the following:

JOBNUM	STATE	IPRI	JIN	JLIST	INTRODUCED	JOB NAME
#J6546	EXEC		10S	LP	THU 12:42A	TRNSPOOL,MGR.NSD
#J6539	EXEC		10S	PP	THU 12:32A	SPOOLJ,UNISPOOL.SYS
#J6540	EXEC		10S	LP	THU 12:41A	JINETD.NET.SYS

```
3 JOBS (DISPLAYED):
```

```
0 INTRO
0 WAIT; INCL 0 DEFERRED
3 EXEC; INCL 0 SESSIONS
0 SUSP
```

```
JOBFENCE= 6; JLIMIT= 10; SLIMIT= 60
```

```
CURRENT: 1/15/96 16:12
```

JOBNUM	STATE	IPRI	JIN	JLIST	SCHEDULED-INTRO	JOB NAME
#J6667	SCHED	15	10S	PP	1/15/96 16:50	CHECKJOB,MANAGER.SYS

```
1 SCHEDULED JOB(S)
```

2. Issue the ABORTJOB command, specifying JINETD's job number on the command line. For example, if JINETD were logged on as job number "6540", you would enter:

```
:ABORTJOB #J6540
```

NOTE If you have started inetd interactively, you use the -k option to kill (stop) it. To do so, enter INETD.NET.SYS -k at the CI prompt or enter /etc/inetd -k from the POSIX shell.

Summary of inetd Command Line Options

There are three options that you may add to the command line when you enter INETD.NET.SYS at the MPE CI prompt or enter /etc/inetd from the POSIX shell.

- c Instructs inetd to reread the configuration file. Use this after you have made changes to the configuration (such as adding a new service) that you want to put into effect now, for an executing inetd.
- k Kills, or stops, the currently executing inetd.
- l A toggle command that starts or stops connection logging for inetd.

Using inetd Message Logging

There are two kinds of message logging that you, as System Manager, can use to monitor and manage Internet Services on your system. The first type is event logging, which is always enabled. It records informational messages, error messages and warnings about the Internet Services. The second type is connection logging, which you can enable and disable. It records successful and failed connection attempts and its own status (on or off). Both event logging and connection logging write messages to the `$STDLIST` device for `inetd` and, in some cases, to the system console.

The kinds of informational, error, and warning messages that are always reported for `inetd`, and what they mean, are listed in the “Troubleshooting” section, later in this chapter. Connection logging is explained next.

Connection Logging

When connection logging is enabled, the Internet daemon records both successful and failed attempts to establish a connection with the host system you are managing. Reviewing the log file can give you important information for managing the Internet Services on your system including:

- Which services are heavily used and which are not.
- Identity of the clients using the Internet Services on your system.
- Pattern of usage, daily, weekly or monthly, for example, for a particular service or set of services.
- Which host(s) are being used for unsuccessful connection attempts, which can indicate who may be attempting to access to your system without authorization.

The syntax of the messages you will see appears here:

```
<<server>><<protocol>><<user>><<program>>  
<<status>>:<<error-msg>>
```

Enable and Disable Connection Logging

The same command turns connection logging on or off, depending upon its current state. So, for example, if message logging is currently disabled, enter the following command at the CI prompt to turn it on:

```
:INETD.NET.SYS -1''
```

Or, from the POSIX shell, enter the following command:

```
$/etc/inetd -1
```

If message logging is enabled, use either the CI or POSIX command shown above to turn it off.

Troubleshooting inetd

This section explains the kinds of error messages you may see regarding the operation of `inetd`. The messages will appear either on the console or they will be sent to the `$STDLIST` for `inetd` or both, depending upon the message's level of importance.

Message	Explanation
An <code>inetd</code> is already running	You attempted to start <code>inetd</code> when one is already running. You may invoke <code>inetd</code> a second time if you use the <code>-c</code> , <code>-k</code> , or <code>-l</code> option, but you cannot run multiple copies of <code>inetd</code> .
There is no <code>inetd</code> running	You attempted to reconfigure <code>inetd</code> when none was running. The first time you run <code>inetd</code> , you must stream it as a job or run it interactively without specifying the <code>-c</code> (reconfiguration) option.
Inetd not found	This message occurs if you invoke <code>inetd</code> with the <code>-c</code> option and <code>inetd</code> cannot reread its configuration file (which is the purpose of <code>-c</code>). This occurs when the original Internet daemon dies or is killed without releasing its semaphore. (The Internet daemon locks a global semaphore to indicate when it is running to prevent users from running more than one <code>inetd</code> at a time.) To fix the problem, enter the <code>inetd -k</code> command to remove the semaphore left by the previous Internet daemon, then restart <code>inetd</code> .

The following diagnostic messages are generated by successful and failed attempts to establish a connection to the Internet Services.

Message	Explanation
<code>/etc/inetd.conf:</code> Unusable configuration file	The Internet daemon cannot access its configuration. The error message preceding this one specifies the reason for the failure.
<code>/etc/inetd.conf:</code> line number: <i>nnn</i> error	There is an error on the line specified by <i>nnn</i> in the <code>inetd</code> configuration file. The Internet daemon skips this line, continues reading the rest of the file, and configures itself accordingly. To solve the problem, open the configuration file, edit the erroneous line, and save the corrected version. Then, tell <code>inetd</code> to reread the new version of <code>INETDCNF</code> by issuing the <code>inetd.net.sys -c</code> command at the CI prompt.
system call:...	The system call noted in the error message failed. See the corresponding entry in the <i>Berkeley Sockets/iX Reference Manual</i> for a description of the system call. The reason for the failure is explained in the error message appended to the system call name.
Cannot configure <code>inetd</code>	Due to errors in the <code>inetd</code> configuration file, none of the services it lists could be set up properly.
Too many services running	The maximum number of services allowed to access <code>inetd</code> simultaneously has been exceeded.

Message	Explanation
file: found before end of the line	An entry in a configuration file may need to exceed one line. If so, you indicate that the line continues by inserting a backslash at the end, then continue typing data on the next line. If, however, you place a backslash in the middle of the line, <code>inetd</code> will ignore it and continue reading to the end of the current line, but will not continue to the next line. In this case, it is likely that the configuration information will be misread.
service/protocol: Unknown service	The system call <code>getservbyname</code> failed because the service is not listed in the services file. To solve the problem, you may either add an entry for the service to the services file or delete the entry for the service from the <code>inetd</code> configuration file.
service/protocol: Server failing (looping), service terminated.	When <code>inetd</code> tries to start 40 servers within 60 seconds for a datagram service, it assumes that the server is failing to handle the connection. To avoid entering a potentially infinite loop, <code>inetd</code> issues this message, discards the packet requesting the socket connection, and refuses further connections for this service. After 10 minutes, <code>inetd</code> tries to reinstate the service and accept connection requests.
service/protocol: socket service/protocol: listen service/protocol: getsockname	Any of these three errors renders the service unusable. To make the service available again, you must issue the <code>inetd -c</code> command to have <code>inetd</code> reread the configuration file.
service/protocol: bind:...	Indicates that the service is temporarily unusable because <code>inetd</code> cannot bind the service to the socket. After 10 minutes, <code>inetd</code> tries to bind the socket again. If it is successful, then it will listen for a connection request and provide the appropriate service. If it fails, it will wait another 10 minutes and try again.
service/protocol: Access denied to remote host (address)	The remote host failed to pass the security test for the service indicated in the message. If this message appears frequently, it can indicate that someone is trying to repeatedly access your system, and failing.
service/protocol: Connection from remote host (address)	When connection logging is enabled, this message indicates a successful connection attempt to the specified service.
service/protocol: Added service, server executable	Records the services that are added when you reconfigure <code>inetd</code> .
service/protocol: New...	Lists the new user identifications, new servers, or executable programs used for the service when reconfiguring <code>inetd</code> .
service/protocol: Deleted service	Records the services that are deleted when you reconfigure <code>inetd</code> .

The following diagnostic and error messages are generated by problems in the `inetd`

security file.

Message	Explanation
<code>/usr/adm/inetd.sec:</code> Field contains other characters in addition to * for service	The wildcard character (*) is used in combination with additional integer(s) in one part of an address field, which is not allowed. For example, the Internet address 10.5*.8.7 entered in the <code>inetd</code> security file will generate an error message because the second field includes a 5 followed by the * character. Either integers or the wildcard character is allowed in one part of an address field.
<code>/usr/adm/inetd.sec:</code> Missing low value in range for service	You have used the range indicator (-) in the wrong way in an entry in the <code>inetd</code> security file. For example, the second field of the Internet address 10.-5.8.7 is incorrect because it does not include both a starting range number ("high value") and the ending range number ("low value"). A correct use of the range indicator in an Internet address would be 10.8-5.8.7.
<code>/usr/adm/inetd.sec:</code> Missing high value in range for service	You have used the range indicator (-) in the wrong way in an entry in the <code>inetd</code> security file. For example, the second field of the Internet address 10.5-.8.7 is incorrect because it does not include both a starting range number ("high value") and the ending range number ("low value"). A correct use of the range indicator in an Internet address would be 10.8-5.8.7.
<code>/usr/adm/inetd.sec:</code> High value in range is lower than low value for service	You expressed a range of numbers incorrectly in an entry in the <code>inetd</code> security file. For example, the second field of the Internet address 10.5-8.8.7 is incorrect because the starting range number ("high value") is lower than the ending range number ("low value"). A correct use of the range indicator in an Internet address would be 10.8-5.8.7.
<code>/usr/adm/inetd.sec:</code> allow/deny field does not have a valid entry for service.	The entry in the second column is not one of the keywords <code>allow</code> or <code>deny</code> . The <code>inetd</code> server ignores the entry and does not implement security for this service unless there is a subsequent entry in the <code>inetd</code> security file for this service that is correct.

Implementation Differences

The implementation of `inetd` on the HP 3000 differs from `inetd` on the HP 9000 in the following ways:

- On the HP 3000, you normally run `inetd` as a job.
- On the HP 3000, there is no `syslogd` server. Instead, all error and informational messages about `inetd` are automatically written to `$STDLIST` for `inetd`. When you run `inetd` as a job, messages are sent to the job's output spool file. Messages which would be logged at the `syslogd` warning log level on HP-UX are, on MPE/iX, additionally sent to the console.

3 Telnet Service

With the release of version C.55.00 of MPE/iX, Telnet server functionality is available to HP 3000 customers. The Telnet server allows users on a remote system that supports the **TCP/IP** and Telnet protocols to log on and run applications on the HP 3000. The Telnet client, which was first made available on version C.50.00 of MPE/iX, gives users on an HP 3000 direct access to other systems that support Telnet and TCP/IP.

This chapter describes:

- How to verify the installation of the Telnet files
- How to edit the `inetd` configuration file and the `services` file to configure the Telnet server.
- How to start the Telnet server once the product has been configured.
- How to troubleshoot problems that arise with Telnet
- Implementation differences between `Telnet` for MPE/iX and `Telnet` for HP-UX.

Before release C.55.00, the capability to receive incoming Telnet connections on the HP 3000 was only available with DTC Telnet access. The HP 3000 processed such connections via a DTC configured with a Telnet Access Card (TAC) using PC-based management software. HP 3000 customers can continue to use DTC Telnet access, particularly if the level of Telnet traffic places a heavy load on the processing capacity of the host HP 3000's CPU.

NOTE Online information about the Telnet client and server is available in the ASCII file `TELNTDOC.ARPA.SYS`.

Overview of Telnet Service

Telnet service consists of a Telnet client and a Telnet server.

The Telnet server uses the standard virtual terminal protocol, originally developed by the Advanced Research Projects Agency (ARPA) to allow users on a remote node that supports the Telnet and TCP/IP protocols to log on and run applications on the host HP 3000. When you configure and enable Telnet on your system, `inetd`, the master server for the Internet Services, will listen for connection requests from Telnet clients. If the request comes from an authorized client node (for example, one that is allowed Telnet access to the host via the `allow` entry in the `inetd` security file), `inetd` will accept the request and start a Telnet session for the requesting client.

The Telnet client allows users on your system to log onto and run applications on a remote host system that supports Telnet access. On MPE/iX, the Telnet client is the program file `TELNET.ARPA.SYS`.

Read “Implementation Differences” for a discussion of the differences between the implementation of the Telnet server on the HP 3000 and the Telnet server as it is implemented on HP-UX systems.

Verifying Installation of Telnet Files

If you have installed or updated to version C.60.00 of MPE/iX, use the following steps to verify that the Telnet software exists on your system:

1. If necessary, log on the system as `MANAGER.SYS`.
2. Run `NMMAINT` to verify that you have successfully installed the Telnet files.

```
:NMMAINT,72
```

You will see information similar to the following.

```
NMS Maintenance Utility 32098-20014 B.00.09 (C) Hewlett Packard Co. 1984
```

```
THU, JAN 18, 1996, 1:39 PM
```

```
Data comm products build version: N.55.08
```

```
Subsystem version ID's:
```

```
HP TELNET/iX Subsystem HP32040A module versions:
```

```
NM program file: TELNET.ARPA.SYS      Version: A5500000
NL procedure:    PTD_SM_VER           Version: A5500000
NL procedure:    PTD_HANDLER_VER      Version: A5500002
NL procedure:    PTD_PTID_VER        Version: A5500001
NL procedure:    PTD_PTOD_VER        Version: A5500001
NL procedure:    PTD_COMMON_VER      Version: A5500000
```

```
HP TELNET/iX Subsystem HP32040A overall version = A.55.00
```

3. Check the final line of the display to make sure there are no error messages such as a module is missing or there is a version mismatch. For example:

```
Version levels differ in one or more modules. (NMERR 103)
```

```
HP TELNET/iX Subsystem HP32040A overall version = ?.??.??
```

4. Issue a `LISTGROUP` command for `ARPA.SYS` to verify that its capabilities are `PM`, `PH`, `IA`, and `BA`.
5. Issue a `LISTF` command for the Telnet files in `ARPA.SYS` to verify that ANY (anyone) can read `TELNTDOC.ARPA.SYS` and that ANY (anyone) can read and execute `TELNET.ARPA.SYS`. Enter:

```
:LISTF TEL@.ARPA.SYS,3
```

Configuring Telnet Server

To configure Telnet, you will edit two files: the services file, which lists the individual services that comprise the suite of Internet Services, and the `inetd` configuration file, which informs the Internet daemon about running Telnet on this system.

Editing the Services File

The services file associates official service names and aliases with the port number and protocol the services use. To enable Telnet, you must edit the services file. Perform the following:

1. Open the services file with an MPE text editor. You may edit the `/etc/services` file from the POSIX shell or the `SERVICES.NET.SYS` file from MPE/iX, whichever you prefer. Both file names should point to the same file.
2. Verify that the following line exists in the file or add it if it does not:

```
telnet 23/tcp
```
3. If the line already exists in the file and it is preceded by a pound symbol (`#`), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.

For more detailed information about editing this file, read Chapter 1 , “Introduction to Internet Services.”

Adding Telnet Service to inetd Configuration

The configuration file for `inetd` determines which installed Internet Services are available to users. To add Telnet service to your system, you need to edit the configuration file for `inetd`, then have `inetd` re-read the configuration. Perform the following:

1. Open the configuration file with a text editor. You may edit the `/etc/inetd.conf` file from the POSIX shell or the `INETDCNF.NET.SYS` file from MPE/iX, whichever you prefer. Both file names should point to the same file.
2. Verify that the following line exists in the file or add it if it does not:

```
telnet stream tcp nowait MANAGER.SYS internal
```
3. If the line already exists in the file and it is preceded by a pound symbol (`#`), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.
5. Signal `inetd` to reread the configuration file by entering the following command at the CI prompt:

```
:INETD.NET.SYS -c
```

Or you may enter this command from the POSIX shell:

```
$/etc/inetd -c
```

6. If you have added the Telnet server to the `inetd` configuration file while the Internet daemon is not running, you must start `inetd` to start the Telnet server. To do so, stream the job `JINETD.NET.SYS` from the CI prompt.

```
:STREAM JINETD.NET.SYS
```

For more detailed information about editing this file, read Chapter 2 , “Internet Daemon.”

Troubleshooting Telnet

This section explains the kinds of errors that may arise regarding the operation of Telnet. The Telnet client user will, in all but one case, be alerted about the problem directly; an error message will appear on the client's terminal. You, as system manager of the host system may receive phone calls from client asking you to investigate the problem.

Problem	Explanation
Unknown service	This message will be written to <code>\$STDLIST</code> for <code>JINETD.NET.SYS</code> when a Telnet client is unable to find the Telnet entry in the services file. Telnet client users may see a similar message on their terminal, and call you, the system manager of the host, to resolve the problem. Open the services file and make sure that the line <code>telnet 23/tcp</code> exists. If necessary add the line and then reconfigure the Internet daemon. For more information, read "Editing the Services File" earlier in this chapter.
The Telnet client cannot run Telnet	The Telnet client user may not have entered the correct program name at the prompt, which is <code>TELNET.ARPA.SYS</code> . Or, there may be problems with the network.
The Telnet client cannot connect to the host	The Telnet client user can encounter this problem for one of several reasons: <ul style="list-style-type: none">• The user entered the domain name, IP address, or NS node name incorrectly.• The system the client attempted to access does not support Telnet.• The network of the system the client attempted to access is not working.• The Internet daemon is not running on the system the client tried to access.
There is a host name lookup failure	The Telnet user tried to log on when the network was not running. Or the host system the client tried to access is not configured on the network.
The Telnet client cannot logon to a host	The Telnet client successfully established a connection to the host, but could not logon. The user may call you, as host system manager, to verify that the logon account and passwords are correct and to see if the system limits are set such that new Telnet sessions are prohibited.

Problem	Explanation
The Telnet server cannot run an application	The Telnet client successfully established a Telnet connection and logs on to the host system. But, when the user runs the application, the software behaves oddly or it produces error messages. If you receive a call about this problem, you or the user can consult the <i>Asynchronous Serial Communications Programmer's Reference Manual</i> to see if the application is attempting to use file system intrinsics that the Telnet server doesn't support. Or have the user and his or her system manager check the set and toggle values on their system to make sure they are the values required by the application.
Invalid command	The Telnet client user entered an invalid command at the Telnet prompt. Type a question mark (?) to display a list of valid commands.

Implementation Differences

The implementation of Telnet on the HP 3000 does not use a separate `telnetd` server file similar to the `tftpd` or `bootpd` server. Instead, Telnet server functionality is provided by code that resides in `NL.PUB.SYS` on version `C.60.00` of MPE/iX. As a result, the last column of the Telnet entry in the `inetd` configuration file is the word “internal.” For example:

```
telnet stream tcp nowait MANAGER.SYS internal
```

By contrast, the entry for the `BOOTP` server in the `inetd` configuration file shows “`bootpd`” in the last column because the `BOOTP` server is not implemented internally. For example:

```
bootps dgram udp wait MANAGER.SYS /SYS/NET/BOOTPD bootpd
```

The implementation of the Telnet server as an internal program concerns you as system manager, in the following two ways:

- When you issue a `LISTFILE` command for `NET.SYS`, you will not see a `telnetd` server file. You do, however, edit the `services` file and the `inetd` configuration file to enable Telnet on your system as you do for the other Internet Services.
- Any security checking the host does before it initiates a Telnet session for the requesting client must be handled by the Internet daemon’s internal security. Specifically, this means that system programmers cannot write “wrappers,” programs that wrap around the Telnet entry in the configuration file to force a separate security-checking program to run on that socket to determine if the connection can or should be established. Instead, you use the `inetd` security file to allow or deny specific nodes Telnet access to your system. For information, read Chapter 2 , “Internet Daemon.”

4 BOOTP Service

The Internet Boot Protocol daemon, or `bootpd`, is used to boot LAN devices such as routers, printers, X-terminals, and diskless workstations. Nodes on the network use `bootpd` to get configuration information such as an **IP address** and a **subnet mask** and automatically boot the device. This chapter describes:

- How to configure `bootpd`.
- How to start `bootpd` once it has been configured.
- Implementation differences between `bootpd` for MPE/iX and `bootpd` for HP-UX.

Overview of bootpd

The Bootstrap Protocol BOOTP allows a client system to get boot information such as its own IP address, the address of a BOOTP server, and the name of the file it needs to load into its memory and execute to boot the printer. The bootstrap operation happens in two phases. In the first phase, the BOOTP daemon `bootpd` determines the address of a BOOTP server and selects a boot file. In the second phase, the Trivial File Transfer Protocol daemon `tftpd` transfers the boot file to the node that requests it.

bootpd Files

There are three files that you will need to configure and use `bootpd` on your system. These files were copied to the NET group of the SYS account when you installed or updated to version C.55.00 or later, of MPE/iX. Table 4-1. briefly describes each one.

Table 4-1. Files for bootpd

File	Description
BOOTPD.NET.SYS	The program file for <code>bootpd</code> which is linked to the POSIX file <code>/etc/bootpd</code> .
BPTABSMP.NET.SYS	The sample configuration file for <code>bootpd</code> that contains information about all of the network devices this system can boot. You will copy this file to <code>BOOTPTAB.NET.SYS</code> , create a symbolic link from the POSIX file <code>/etc/bootptab</code> to this file, and edit it as necessary.
BOOTPQRY.NET.SYS	A program for testing <code>bootpd</code> . You will not need to copy or edit this file, but you will create a symbolic link from the POSIX file <code>/etc/bootpquery</code> to <code>BOOTPQRY.NET.SYS</code> .

Configuring bootpd

To configure `bootpd`, you will edit three files: the services file, which lists the individual services that comprise the suite of Internet Services, the `inetd` configuration file, which informs the Internet daemon about running `bootpd` on this host, and the `bootpd` configuration file, which contains client and **relay** information. These tasks are explained in the following sections.

Editing the Services File

The services file associates official service names and aliases with the port number and protocol the services use. To enable `bootpd`, you must edit the services file. Perform the following:

1. Open the services file with an MPE text editor. You may edit the `/etc/services` file from the POSIX shell or the `SERVICES.NET.SYS` file from MPE/iX, whichever you prefer.

Both names should point to the same file.

2. Verify that the following lines exist in the file or add them if they do not:

```
bootps 67/udp # Bootstrap protocol server
bootpc 68/udp # Bootstrap protocol client
```

3. If the lines already exist in the file and they are preceded by a pound symbol (#), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.

Adding BOOTP Server to inetd Configuration

The configuration file for `inetd` determines which installed Internet Services are available to users. To add `bootpd` to your system, you need to edit the configuration file for `inetd`, then have `inetd` re-read the configuration. Perform the following:

1. Open the `inetd` configuration file with a text editor. You may edit the `/etc/inetd.conf` file from the POSIX shell or the `INETDCNF.NET.SYS` file from MPE/iX, whichever you prefer. Both names should point to the same file.
2. Verify that the following line exists in the file or add it if it does not:

```
bootps dgram udp wait MANAGER.SYS /SYS/NET/BOOTPD bootpd
```

3. If the line already exists in the file and it is preceded by a pound symbol (#), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.
5. Signal `inetd` to reread the configuration file by entering the following command at the CI prompt:

```
:INETD.NET.SYS -c
```

Or you may enter this command from the POSIX shell:

```
$/etc/inetd -c
```

6. If you have added `bootpd` to the `inetd` configuration file while the Internet daemon is not running, you must start `inetd` to start the BOOTP server. To do so, stream the job `JINETD.NET.SYS` from the CI prompt.

```
:STREAM JINETD.NET.SYS
```

For more detailed information about editing this file, read Chapter 2 , “Internet Daemon,”

The bootpd Configuration File

When `bootpd` is started, it reads a configuration file to find out information about clients and relays, then listens for boot request **packets**. By default, `bootpd` uses the configuration file `/etc/bootptab`, but you may specify another configuration file.

The BOOTP server will reread its configuration file and update its information about new, deleted or modified hosts on two occasions other than startup: when you send it a **SIGHUP signal**, or when it receives a boot request packet and detects that the configuration file has been edited.

Creating and Linking bootpd Configuration File

You may already have a configuration file for `bootpd` installed on your system. If you know that you have such a file, and it is accessible by the POSIX file name `/etc/bootptab` you may skip these steps.

If not, follow the steps below to create the file and link to it. If you have such a file, but are unsure whether or not it is linked, perform step 2 only.

1. Create your own configuration file by using the `COPY` command to rename the sample file. Enter:

```
:COPY BPTABSMP.NET.SYS TO BOOTPTAB.NET.SYS
```

2. Create a symbolic link from `/etc/bootptab` in the POSIX name space to `BOOTPTAB.NET.SYS`. Enter:

```
:NEWLINK /etc/bootptab, BOOTPTAB.NET.SYS
```

3. Check the security provisions of the file and change them, if necessary. Hewlett-Packard recommends that only `MANAGER.SYS` has write access to `BOOTPTAB.NET.SYS`, and write and purge access to `/etc/bootptab`.

Editing the bootpd Configuration File

Use the following steps to edit the `bootpd` configuration file:

1. Open the file with an MPE text editor. You may edit the `/etc/bootptab` file from the POSIX shell or the `BOOTPTAB.NET.SYS` file from MPE/iX, whichever you prefer. Both file names should point to the same file.
2. Add, delete, or change any of the entries in the file. The following sections give you more information about the contents of the `bootpd` configuration file.
3. Save the file and exit the editor program.

Adding Client and Relay Data to bootpd Configuration File

To allow a client to boot from your local system or to allow a boot request to be relayed to the appropriate boot server, you must add information about the client to the `bootpd` configuration file. This file contains client entries and relay entries. Client entries provide the information necessary to allow clients to boot from your system. Relay entries provide

the information necessary to relay a boot request to one or more bootpd servers.

The information that you need to collect for these types of entries is explained in the next two sections.

Collecting Client Information

To make an entry for the client in the bootpd configuration file, you need to collect information about the client such as the following:

- Name of the client's system
- Type of network interface hardware (IEEE 802.3 or Ethernet)
- Client's hardware address
- Client's assigned IP address
- IP address mask that identifies the network where the client resides
- Address of the gateway for the client's local subnet
- Name of the boot file that the client will retrieve using TFTP.

Collecting Relay Information

To make a relay entry for the client in the bootpd configuration file, you need to collect information such as the following:

- Name of the client's system
- Type of network interface hardware (IEEE 802.3 or Ethernet)
- Client's hardware address
- Subnet mask used to identify the network address where the client resides
- Address of the gateway that connects the client's local subnet to the intended BOOTP server's subnet
- IP addresses of the BOOTP servers to which the local system will relay the client's boot request
- Threshold value, which is the number of elapsed seconds since the client's first request
- Maximum number of hops that the client's boot request can be forwarded

Syntax of bootpd Configuration Entries

An entry in the bootpd configuration file consists of a single line with the following format:

```
hostname:tag=value tag=value tag=value
```

The `hostname` is the actual name of a BOOTP client and the `tag` is a two-character case-sensitive symbol. Most tags are followed by an equal sign and a value, as shown above, though some tags do not require a value. The BOOTP daemon uses these tags and values to recognize a client's boot request, supply parameters in the bootreply to the client, or relay the boot request.

For example, here is an entry for client `printer01`:

```
printer01: ht=ether: ha=080009030166: ip=15.19.8.2:\\ sm=255.255.248.0:
gw=15.19.8.1: bf=/printer01
```

This entry tells `bootpd` that the host `printer01` uses an Ethernet network interface (`ht=ether`) whose hardware address (`ha`) is `080009030166`. The IP address (`ip`) is `15.19.8.2`, the Subnet mask (`sm`) is `255.255.248.0`, and the address of the gateway (`gw`) is `15.19.8.1`. The bootfile that `tftpd` will transmit to boot this printer (`bf`) is `/printer01`.

Tags Used in bootpd Configuration File

You can use any of the following tags to enter client or relay data into the `bootpd` configuration file.

Tag	Description
<code>ba</code> or <code>ba=address</code>	Tells <code>bootpd</code> to broadcast the boot reply to the client. If you specify no value for <code>ba</code> , <code>bootpd</code> sends the boot reply on the configured broadcast address of each network interface on the server's system. If you specify an IP-address for its value, <code>bootpd</code> sends the boot reply to a specific IP or broadcast address. Use the <code>ba</code> tag only for diagnostic purposes, for example when debugging boot replies with <code>BOOTPQRY</code> .
<code>bf=filename</code>	Specifies the filename, in Hierarchical File Structure (HFS) syntax, of the bootfile that the client should download. The client's boot request, and the values of the <code>hd</code> and <code>bf</code> tags, determine the contents of the bootfile field in the boot reply packet.
<code>bs=size</code> or <code>bs</code>	Specifies the size of the bootfile in 512-octet blocks, expressed as a decimal, octal, or hexadecimal integer. Or, if you omit the value, <code>bootpd</code> will automatically calculate the bootfile size at each request.
<code>ds=ip address list</code>	Specifies the IP address of one or more RFC1034 Domain Name servers.
<code>gw=ip address list</code>	Specifies the IP address of one or more gateways for the client's subnet. If you prefer one of multiple gateways, list it first.
<code>ha=hardware-address</code>	Specifies the hardware address of the client in hexadecimal. You may include periods and/or a leading <code>0x</code> for readability. The <code>ha</code> tag must be preceded by the <code>ht</code> tag either explicitly or implicitly; see <code>tc</code> below.
<code>hd=home-directory</code>	Specifies an HFS directory name to which the bootfile is appended (see <code>bf</code> tag above). The default value is <code>(/)</code> .
<code>hn</code>	Directs <code>bootpd</code> to send the client's hostname in the boot reply. The <code>BOOTP</code> daemon attempts to send the entire hostname as it is specified in the configuration file. If this cannot fit into the reply packet, it attempts to shorten the name to just the host field (up to the first period, if present) and send that. In no case will <code>bootpd</code> send an arbitrarily truncated hostname. If nothing reasonable can fit, it sends nothing.

Tag	Description
ht=hardware-type	Specifies the hardware type code. The hardware-type can be an unsigned decimal, octal, or hexadecimal integer corresponding to one of the ARP Hardware Type codes specified in RFA1010. The HP 3000 implementation will support <code>ether</code> for ethernet networks and <code>ieee802</code> for IEEE 802.3 networks.
ip=ip address	Specifies the IP address of the BOOTP client.
sm=subnet-mask	Specifies the client's subnet mask as a single IP address.
Tnnn=generic-data	A generic tag where <i>nnn</i> is an RFC1048 vendor field tag number. This allows <code>bootpd</code> to immediately take advantage of future extensions to RFC1048. The generic-data data can be represented as either a stream of hexadecimal numbers or as a quoted string of ASCII characters. The length of the generic data is automatically determined and inserted into the proper fields of the RFC1048-style boot reply.
tc=template-host	Indicates a table continuation. Often many host entries share common values for certain tags (such as domain servers) and, rather than repeatedly specifying these tags, a full specification can be listed for one host entry and shared by others. The template-host is a dummy host (configuration file entry) for a host that does not actually exist and never sends boot requests. Information explicitly specified for a host always overrides information implied by a <code>tc</code> tag symbol, regardless of its location within the entry. The value of <code>template-host</code> can be the hostname or IP address of any host entry previously listed in the configuration file. If it is necessary to delete a specific tag after it has been inferred via <code>tc</code> , enter <code>tag@</code> . For example, to undo an RFC1034 domain name server specification, use <code>:ds@:</code> at an appropriate place in the configuration entry. After canceling the tag this way, you may set it again.
to=offset	Specifies the client's time zone offset in seconds from UTC. The time offset can be either a signed decimal integer or the keyword <code>auto</code> which uses the server's time zone offset.
ts=ip_address_list	Specifies the IP address of one or more RFC868 Time Protocol servers.
vm=magic-cookie	Specifies the RFC1048 vendor information magic cookie, <code>magic-cookie</code> can be one of the following keywords: <code>auto</code> , indicating that vendor information is determined by the client's request, <code>rfc1048</code> , which always forces an RFC1048-style reply, or <code>cmu</code> , which always forces a CMU-style reply.

Editing Tips

When you are updating the `bootpd` configuration file, keep the following points in mind:

- Client's hostname must be the first field of an entry.

- If you specify an `ht` tag, it must precede the `ha` and `hm` tags.
- If you specify the `gw` tag, you must also specify the `sm` tag.
- IP addresses listed for a single tag must be separated by a space.
- A single client entry can be extended over multiple lines if you use a backslash (`\`) at the end of each line.
- Blank lines and lines that begin with the pound sign (`#`) are ignored.

A relay entry can contain relay parameters for an individual system or for a group of systems. If a BOOTP client does not have an individual entry in the `bootpd` configuration file, `bootpd` searches the group relay entries and uses the first group relay entry that matches the BOOTP client.

Sample bootpd Configuration Files

The two following examples show sample `bootpd` configuration files.

The first example shows the configuration for a simple network without gateways or subnets.

```
#
#
# The first entry is the template for options common to all of the printers.
#
#global.defaults:\\
#    hn:\\
#    ht=ether:\\
#    vm=rfc1048:\\
#
# Now the actual entries for the individual printers are listed.
#
#printer1:\\
#    tc=global.defaults:\\
#    ha=08000903212F:\\
#    ip=10.13.193.72
#
#printer2:\\
#    tc=global.defaults:\\
#    ha=0800090324AC:\\
#    ip=10.13.193.73
#
#
```

The second example shows the configuration for a network with gateways and subnets.

```
#
#
#printer1:\\
#    tc=global.defaults:\\
#    ha=08000903212F:\\
#    gw=10.13.192.2:\\
#    sm=255.255.248.0:\\
#    ip=10.13.193.72
#
#printer2:\\
#    tc=global.defaults:\\
```

```
#      ha=0800090324AC:\\
#      gw=10.13.192.2:\\
#      sm=255.255.248.0:\\
#      ip=10.13.193.73
#
```

Starting bootpd

To successfully start `bootpd`, you must have a current and correct configuration file for it. The default file is `/etc/bootptab` but you may use an alternate configuration file by specifying its POSIX file name on the command line. Without this configuration file, `bootpd` will not be able to service BOOTP requests.

You can run `bootpd` under the Internet daemon only. You may not run it as a standalone server.

Starting bootpd Under inetd

If you are running `bootpd` with `inetd`, make certain that you have edited the `inetd` configuration file as explained earlier in this chapter. There is no special step required of you to start `bootpd`: When the Internet daemon is running, it will automatically invoke `bootpd` when it gets a connection request for that service. To find out how to start `inetd`, refer to Chapter 2, “Internet Daemon.”

Command Line Options for bootpd

You can change the way that `bootpd` operates by entering the `bootpd` command followed by one of the command line options. For example:

```
:BOOTPD.NET.SYS -d
```

The options available to you are explained below.

Option	Purpose
<code>-t</code>	Changes the timeout value for <code>bootpd</code> . The BOOTP daemon starts when the first BOOTP request arrives. If no other boot request arrives within the default period of 15 minutes, <code>bootpd</code> ends. If you specify a timeout of 0 minutes, the server will not die until you abort <code>JINETD</code> or <code>JINETD</code> ends in an error state.
<code>-d</code>	Sets the verbosity level for the logging messages generated by <code>bootpd</code> .
<code>configfile</code>	The configuration file <code>bootpd</code> reads to get configuration information, expressed in HFS syntax. By default, <code>bootpd</code> uses <code>/etc/bootptab</code> .

Troubleshooting bootpd

The `BOOTPQRY` program is a diagnostic tool used to check the configuration of `bootpd`. It uses the supplied parameters to construct a boot request to send to a BOOTP server. It prints the contents of the boot reply, including the client's Internet address, the name of a boot file, and the name and address of the server that sent the reply. `BOOTPQRY` formats and prints RFC1048 or CMU-style vendor information included in the reply.

The boot request packet is broadcast on the BOOTP server port. Responding servers return a bootreply packet on the BOOTP client port. `BOOTPQRY` can only display bootreply packets when the BOOTP server broadcasts the reply on the client port or when the hardware address and IP address supplied in the boot request are those of the host on which `BOOTPQRY` is run.

To use the `BOOTPQRY` program to troubleshoot `bootpd`, do the following:

1. Open the `bootpd` configuration file and look for the entry describing the network device you want to test.
2. When you find the entry, add the `ba` tag to it. This will force `bootpd` to broadcast the reply so that `BOOTPQRY` can display it.
3. Run the `BOOTPQRY` program by entering the `BOOTPQRY` command followed by the hardware address of the network you are testing, expressed in hexadecimal notation. For example, at the CI prompt you would enter:

```
:BOOTPQRY.NET.SYS 08000902CA00
```

Or, from the POSIX shell, you would enter:

```
$/etc/bootpquery 08000902CA00
```

Diagnostic Options

The following options provide the information for the boot request:

Option	Purpose
<code>haddr</code>	The hardware address of the BOOTP client to use in the boot request. A BOOTP server responds if it has configuration information for a host with this link level address.
<code>htype</code>	The type of address specified as <code>haddr</code> , which may be <code>ether</code> or <code>ieee802</code> . The default address type is <code>ether</code> .
<code>-i<ipaddr></code>	The Internet address of the BOOTP client <code><ipaddr></code> to use in the boot request. If the BOOTP client doesn't know its IP address, the BOOTP server supplies it in the bootreply. Otherwise, the server returns the bootreply directly to <code>ipaddr</code> .
<code>-s<server></code>	The name of the BOOTP server <code><server></code> to which the boot request should be sent directly. When the BOOTP server is known, the boot request is not broadcast.
<code>-v<vendor></code>	Request vendor information for <code><vendor></code> . The vendor can be specified as

rfc1048 or CMU. For any other vendor specification, the first four characters of the parameter are used as the vendor magic cookie.

-f<bootfile> Specify a boot file needed by the BOOTP client. If a boot file is specified in the boot request, the BOOTP server responds only if the server host can make the file available via TFTP.

Sample Diagnostic Results

Here is an example of BOOTPQUERY output:

```
# bootpquery 0800092175ff

Received BOOTREPLAY from hpmpe992.cup.hp.com (15.19.134.20)
hardware Address: 08:00:09:21:75:ff
Hardware Type ethernet
IP Address: 15.19.123.53
Boot file: (None)

RFC1048 Vendor Information:
Subnet Mask: 255.255.248.0
Log Server 15.19.134.20
Host Name; hpljnet2
Tag #144 [104, 112, 110, 112, 108, 106,
         110, 101, 116, 46, 99, 102, 103]
```

Implementation Differences

The implementation of `bootpd` on the HP 3000 differs from `bootpd` on the HP 9000 in following ways:

- The BOOTP entry in the `inetd` configuration file must have an MPE/iX compatible user name. Hewlett-Packard recommends that you use `MANAGER.SYS`.
- You cannot run `bootpd` as a standalone server. It can only be run by the Internet daemon.

5 TFTP Service

The Trivial File Transfer Protocol (TFTP) is a basic communications protocol used to transmit files between nodes on a network. It is implemented on top of the Internet User Datagram Protocol (UDP), so it can be used across networks that support UDP. On the HP 3000, the TFTP daemon `tftpd` transfers boot files to or from the host HP 3000 to remote nodes on the network. This permits a network device to get the information it needs to start itself.

This chapter describes:

- How to configure `tftpd`
- How to start `tftpd` once the server has been configured.
- Implementation differences between `tftpd` for MPE/iX and `tftpd` for HP-UX.

Overview of tftpd

TFTP is a simplified version of the File Transfer Protocol (FTP). The primary function of the TFTP daemon `tftpd` is to support the Bootstrap Protocol `BOOTP`, which allows network devices to get the information they need to boot, or start, themselves. Network devices commonly use TFTP to transmit boot files because TFTP is simple enough to be implemented in ROM.

On the HP 3000, the TFTP daemon `tftpd` transfers files to or from the host HP 3000 to remote systems or printers. Configuring `tftpd` on your system allows you to make boot files (and other kinds of files) available to remote clients that support TFTP.

Configuring tftpd

To configure `tftpd`, you will edit two files: the services file, which lists the individual services that comprise the suite of Internet Services, and the `inetd` configuration file, which informs the Internet daemon about running `tftpd` on this system. These tasks are explained in the next sections.

Editing the Services File

The services file associates official service names and aliases with the port number and protocol the services use. To enable `tftpd`, you must update the services file. Perform the following:

1. Open the services file with an MPE text editor. You may edit the `/etc/services` file from the POSIX shell or the `SERVICES.NET.SYS` file from MPE/iX, whichever you prefer. Both names should point to the same file.
2. Verify that the following line exists in the file or add it if it does not:

```
tftp 69/udp # Trivial File Transfer Protocol
```
3. If the line already exists in the file and it is preceded by a pound symbol (`#`), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.

Adding TFTP Service to inetd Configuration

The configuration file for `inetd` determines which installed Internet Services are available to users. To add `tftpd` to your system, you will need to edit this configuration file, then have `inetd` re-read the configuration. To do so:

1. Open the `inetd` configuration file with a text editor. You may edit the `/etc/inetd.conf` file from the POSIX shell or the `INETDCNF.NET.SYS` file from MPE/iX, whichever you prefer. Both names point to the same file.

2. Verify that the following line exists in the file or add it if it does not:

```
tftpd dgram udp wait USER.TFTP /SYS/NET/TFTPD tftpd
```

3. If the line already exists in the file and it is preceded by a pound symbol (#), delete the symbol and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.

There are two options in the `tftpd` entry, `[user]` and `[path]`, which are explained in the next two sections. For more detailed information about editing the configuration file, read Chapter 2, “Internet Daemon.”

Specifying the TFTP User

The Internet daemon runs `tftpd` as the user specified in the `[user]` parameter of its entry in the `inetd` configuration file. For example, this entry instructs `inetd` to run the TFTP server as `USER.TFTP`:

```
tftpd dgram udp wait USER.TFTP /SYS/NET/TFTPD tftpd
```

Hewlett-Packard recommends that you run `tftpd` this way, and that you use the following steps to create the `TFTP` account and two user identifications, `USER.TFTP` and `MGR.TFTP`, with the appropriate capabilities:

1. If necessary, log onto the system as `MANAGER.SYS` or to another user identity that has been assigned SM capability.
2. Create the `TFTP` account by entering the following command at the CI prompt:

```
:NEWACCT TFTP,MGR;CAP=AM,PH,DS,ND,SF,IA,BA
```

3. Create the new user of the `TFTP` account with a home directory of `TFTPDIR` by entering the following command at the CI prompt:

```
:NEWUSER USER.TFTP;cap=BA,PH,DS;home=TFTPDIR
```

When a client accesses `tftpd` it will first look for the file in the home group `TFTPDIR`.

4. Create the home directory `TFTPDIR` by entering the following command at the CI prompt:
5. Modify the new manager of the `TFTP` account by entering the following command at the CI prompt:

```
:NEWGROUP TFTPDIR.TFTP
```

```
:ALTUSER MGR.TFTP;cap=PH,DS,ND,SF,IA,BA
```

For security reasons, `USER.TFTP` is not assigned ND, SF, PM or SM capabilities. This way `USER.TFTP` can be used to run `tftpd` while `MGR.TFTP`, who is assigned some of these capabilities, can control which files are placed in the `TFTPDIR` group.

Specifying a Search Path

As an option, you can use the `[path...]` parameter in the `inetd` configuration file entry to specify the list of files or directories that are available to `TFTP` clients. For example, if you would like to have the `/tmp` and `/bin` directories available to `TFTP` clients in addition to the home group of the `TFTP` user, edit the line to look like this:

```
tftp dgram udp wait USER.TFTP /SYS/NET/TFTPD tftpd /tmp /bin
```

When a file is requested by a TFTP client, `tftpd` first looks for a file relative to the home directory of the user specified in the `inetd` configuration file. If it does not find the file there, it then checks to see if the following two conditions are met:

- File requested is at or below `[path]`.
- User specified in the `inetd` configuration file (in the previous examples, `USER.TFTP`) has access to the file.

When invoked with no path arguments, `tftpd` cannot follow symbolic links that refer to paths outside of the home directory of the user specified in the `inetd` configuration file.

Permission to Retrieve Files

If permission is given to remote systems to retrieve a file through TFTP, then the file must be readable by the user specified in the `inetd` configuration file. If permission is given to remote systems to transmit a file through TFTP, then the file must already exist and be writable by the user specified in the `inetd` configuration file.

Starting tftpd

The TFTP daemon runs under the Internet daemon. If you have just added `tftpd` to the `inetd` configuration, you must reconfigure `inetd` to begin using TFTP. To reconfigure `inetd`, enter the following command at the CI prompt:

```
:INETD.NET.SYS -c
```

Or, from the POSIX shell, enter this command:

```
$/etc/inetd -c
```

If you have added `tftpd` to the `inetd` configuration file while the Internet daemon is not running, you must start `inetd` to start the TFTP server. To do so, stream the job `JINETD.NET.SYS` from the CI prompt.

```
:STREAM JINETD.NET.SYS
```

Troubleshooting tftpd

The following error messages may be generated by TFTP and logged with the syslog facility, if it is enabled.

Message	Explanation
Unknown option ignored	An invalid option was specified in the <code>tftpd</code> arguments. Remove or correct the arguments and restart <code>tftpd</code> .
Invalid total time-out	The value given for the <code>-T</code> option was either not a number or was a negative number. Correct the value and restart <code>tftpd</code> .
Invalid retransmission time-out	The value for the <code>-R</code> option was either not a number or was a negative number. Correct the value and restart <code>tftpd</code> .
system call<\$Isystem call>:...	The system call specified in the message failed. The reason for failure is explained in the error message appended to the system call name in its documentation.

Implementation Differences

The implementation of `tftpd` on the HP 3000 differs from `tftpd` on the HP 9000 in three ways:

- On HP-UX, `tftpd` is usually run as `root`. On MPE/iX, it is usually run as `USER.TFTP`.
- On HP-UX, `tftpd` checks if the user `tftp` can write to or read the file. On MPE, `tftpd` checks if the user specified in its configuration file can write to or read the file. If you configure `tftpd` as recommended in this chapter, `USER.TFTP` will be specified in the configuration file and `tftpd` will check the same user.
- On MPE/iX, the `tftp` user is configurable and it is not on HP-UX. As a result, on MPE/iX `tftpd` looks at the file relative to the home directory of whichever user is specified in the `inetd` configuration file. On HP-UX, `inetd` always looks at the file relative to the home directory of the `tftp` user.

6 REMSH Service

The remote shell, or `remsh`, service is used to connect to a specified host and execute a command on that remote host. The remote shell or `remsh` is available with version C.60.00 of the MPE/iX operating system.

This chapter describes:

- How to configure the services file to allow `remsh` to run.
- How to verify that `remsh` is available on the system.
- How to run `remsh`
- Implementation differences between `remsh` on MPE/iX and `remsh` for HP-UX.

Overview of remsh Service

The remote shell `remsh`, is the same service as `rsh` on BSD UNIX systems. The name was changed due to a conflict with the existing command `rsh` (restricted shell) on System V UNIX systems.

Use `remsh` to connect to the remote system and execute a command on that remote system. Output from the remote command is sent to standard output for `remsh`, so the user can see the results of the command.

Verifying Installation of remsh Files

The `remsh` client is part of the Internet Services product with release C.60.00. To verify that `remsh` is available on your system you may use `NMMAINT` verify versions of the Internet services product.

```
hawaii(PUB); nmmaint,73
NMS Maintenance Utility 32098-20014 B.00.09 (C) Hewlett Packard Co. 1984

WED, JUL 23, 1997, 11:08 AM Data comm products build version: N.55.15

Subsystem version ID's:
```

Internet Services for the HP3000 module versions:

NM program file:	INETD.NET.SYS	Version:	B0001003
NM program file:	BOOTPD.NET.SYS	Version:	B0001003
NM program file:	BOOTPQRY.NET.SYS	Version:	B0001002
NM program file:	TFTPD.NET.SYS	Version:	B0001002
NM program file:	REMSH.NET.SYS	Version:	B0001003
XL procedure:	INSVXL_SECURE_VERS	Version:	B0001004
XL procedure:	INSVXL_IPCSEC_VERS	Version:	B0001002
XL procedure:	INSVXL_NSRW_VERS	Version:	B0001003
XL procedure:	INSVXL_NETOF_VERS	Version:	B0001002
XL procedure:	INSVXL_SYSLOG_VERS	Version:	B0001003
XL procedure:	INSVXL_SIGNAL_VERS	Version:	B0001002
XL procedure:	INSVXL_GETTIME_VERS	Version:	B0001003

Internet Services for the HP3000 overall version = B.00.01

Configuring remsh Client

There is only one file on the MPE/iX system that you will need to change in order to allow use of the `remsh` client. That is the file `SERVICES.NET.SYS`. However, there are some files that will need to be configured on the remote UNIX systems.

Editing the Services File

The services file associates official service names and aliases with the port number and protocol the services use. To enable `remsh`, you must edit the services file. Perform the following:

1. Open the services file with a text editor. You may edit the `/etc/services` file from the POSIX shell or the `SERVICES.NET.SYS` file from MPE/iX, whichever you prefer. Both names should point to the same file.
2. Verify that the following line exists in the file, or add it if it does not:

```
shell          514/tcp cmd          # remote command, no passwd used
```
3. If the line already exists in the file and is preceded by a pound symbol (`#`), delete the `#` and any spaces before the service name to enable the service.
4. Save the file and exit the editor program.

UNIX Configuration

The `remsh` service does not prompt for user ID and passwords. That information is handled via the command line parameters and configuration on the UNIX host. See the “Using `remsh`” section for details on how the user id is determined and passed to the UNIX host.

Password information is bypassed by use of a `.rhosts` in the remote user’s home directory or by use of the file `/etc/hosts.equiv`. See the man pages of the UNIX system for details on how to set up a `/etc/hosts.equiv` file. A user’s `.rhosts` file entry will consist of the MPE/iX system name and user id.

If you wish to access the HP-UX Host “`taltos`” as user `cawti` from the MPE/iX system `jhereg` while user `MANAGER.SYS`, you’ll need to set up a host equivalency via the `/etc/hosts.equiv` file, or you will create a `.rhosts` file in the home directory of user `cawti` on the “`taltos`” machine. The `.rhosts` file entry would look like:

```
jhereg MANAGER.SYS
```

This will cause the `remsh` daemon on the UNIX host to allow a connection from `MANAGER.SYS` on `jhereg` to the `cawti` user on the host “`taltos`.” The `.rhosts` file for user `cawti` would contain an entry for every host and userid that you desired to access the “`taltos`” host as if they were the user `cawti`.

NOTE The MPE/iX equivalent of the UNIX user id is the `User.Account`. An artifact of the MPE/iX implementation is that the MPE/iX information is usually reported in upper case. So be sure your `.rhosts` or `/etc/hosts.equiv` entries use the MPE/iX user ID information in uppercase.

Using remsh

The `remsh` service is accessed by running the `REMSH.NET.SYS` program. You may do so under the MPE/iX CI or under the POSIX shell. While the format of the commands will differ depending on how you run the program, the parameter list remains the same.

For the purposes of explaining the parameters, look at a sample invocation from the POSIX shell. Detailed examples of both the POSIX shell and MPE/iX invocations will follow later.

From the POSIX shell, invoke the `remsh` by typing:

```
/SYS/NET/REMSH remotehost -l remoteuser remotecommand
```

In all cases you must provide a `remotehost` and a `remotecommand`. The `remsh` program will fail and generate an error message otherwise. Unless the remote system has MPE/iX type userids, you will also need to provide a `-l remoteuser` parameter as well. Otherwise the remote system will not allow the connection.

The name of the remote host you are attempting to connect to is `remotehost`. The host name can be either the official name or an alias as understood by `gethostbyname()`.

The `userid` is `remoteuser` on the remote system.

NOTE The traditional UNIX implementation of `remsh` makes the `-l remoteuser` parameter optional. If you do not provide a `-l remoteuser` parameter, `remsh` takes your current `userID` and assumes that you wish to connect to the same `userID` on the remote system. Since the MPE version of the `userID` is `USER.ACCOUNT`, and the UNIX equivalent is `user`, it is unlikely that you will find a user on the remote system to match your `id`. We recommend that you always provide the `-l remoteuser` argument to `remsh`.

The `remotecommand` is the command the user wishes to execute on the remote machine. This command may be a CI command, a program (that meets certain criteria) or a shell script. If `remotecommand` is not specified, `remsh` will terminate and provide a usage message.

NOTE `remsh` cannot be used to run commands that require a terminal interface (such as `vi`) or commands that read their standard error (such as `more`).

MPE/iX Examples

To run `remsh` from MPE/iX prompt, type:

```
run remsh.net.sys;info="remotehost -l remoteuser remotecommand"
jhereg(PUB): run remsh.net.sys;info="taltos -l cawti pwd " /u2/home/cawti
END OF PROGRAM
jhereg(PUB):
```

POSIX Examples

From the POSIX Shell prompt, type:

```
/SYS/NET/REMSH remotehost -l remoteuser remotecommand  
shell/iX> /SYS/NET/REMSH taltos -l cawti pwd  
/u2/home/cawti  
shell/iX>
```

There are a number of shell features that can be taken advantage of, while running under the POSIX shell.

Shell metacharacters that are not quoted are interpreted on the local host; quoted metacharacters are interpreted on the remote host. Thus the command line:

```
/SYS/NET/REMSH taltos -l cawti cat remotefile >> localfile
```

appends the remote file remotefile to the local file localfile, while the command line:

```
/SYS/NET/REMSH taltos -l cawti cat remotefile ">>" otherremotefile
```

appends remotefile to the remote file otherremotefile.

The following command line runs remsh in the background on the local system, and the output of the remote command comes to your terminal asynchronously:

```
/SYS/NET/REMSH otherhost -l remoteuser -n remotecommand &
```

The following command line causes remsh to return immediately without waiting for the remote command to complete:

```
/SYS/NET/REMSH otherhost -l remoteuser "remotecommand 1>&- 2>&- &"
```

remsh was written so that if the first parameter in its argument vector is not remsh, it will use the value as a host name. So you may symbolically link the host name to the remsh program. A typical BSD UNIX implementation will have these links under the /usr/hosts directory.

If you have made a symbolic link to the remsh program that is the host name, for example you have already entered, (`ln -s /SYS/NET/REMSH taltos` in our examples), you could simply generate the same result as the first example with the following:

```
shell/iX>taltos -l cawti pwd /u2/home/cawti shell/iX>
```

Troubleshooting remsh

remsh MPE/iX/X version won't support rlogin or rexec functionality usage: remsh host -l login -n command

Be sure to provide a command to execute.

remshd Login incorrect.

Probably invalid entry in remote `.rhosts` file. Be sure host name and user id are correct. User ID must be in uppercase. Be sure you provided a `-l` userid parameter or that the remote system has a userid that matches your MPE/iX logon.

Program requires more capabilities than allowed for the group, the user of a temporary file, or the hierarchical directory user. (LDRERR 505) Native mode loader message 505 Unable to load program to be run. (CIERR 625)

The first message is from running `remsh` from MPE/iX name space and the second from running under the POSIX Shell. The cause is typically lack of PM capability on the group where `remsh` resides. Since `remsh` is in `NET.SYS`, this problem is unlikely to be seen unless, someone changes the capability of the `NET.SYS` group.

```
**** EXEC FUNCTION FAILED; subsys =517; info = 48
ABORT: REMSH.NET.SYS
NM SYS a.00aa0270 dbg_abort_trace+$24 NM UNKN 150.00366f6c NM UNKN 2dd.0004bbd8
[1] +
Done (134) REMSH hpcsyn24 -l casc -n pwd 262204 Abort REMSH
```

shell/tcp Unknown service.

The "shell" service specification is not present in the services file. Edit `/etc/services` or `SERVICES.NET.SYS` to fix.

Can't establish stderr

remsh cannot establish secondary socket connection for stderr.

Couldn't reopen stderr

The remote command tried to reopen `stderr`. This is not allowed under `remsh`.

<system call>: ...

Error in executing system call. Appended to this error is a message specifying the cause of the failure.

Implementation Differences

The full remote shell service typically consists of two parts (the `remsh` client which allows a user on this machine to access remote hosts and the `remshd` server which allows `remsh` clients on other hosts to access the local host). Only the `remsh` client functionality has been implemented on the MPE/iX system.

The UNIX version of the `remsh` client has an optional `-n` parameter that tells the client to not read from `STDIN`. Due to differences between MPE I/O and UNIX I/O the `-n` parameter

has been hard coded into the MPE/iX client.

The HP-UX `remsh` client also allows `rlogin` and `rexec` functionality. Since the MPE/iX implementation was designed to address the needs of users attempting to access UNIX commands/scripts from stream jobs, we chose not to implement any feature needing interactive input with the remote system.

7 Samba/iX Services

Samba/iX is a suite of programs which work together to allow clients to access a server's file space and printers via the Server Message Block (SMB) file server. Samba/iX runs on MPE/iX shell operating system starting in the MPE/iX 6.0 release. It allows the MPE/iX shell operating system to act as a file and printer server for SMB clients, which are, primarily, Windows for Workgroups, Windows 95, Windows NT, and other clients.

Overview of Samba/iX

Samba/iX is a suite of programs which allow an HP 3000 running MPE/iX operating system to provide service using a Microsoft networking protocol called Server Message Block (SMB). This product allows implementation of interoperability features allowing the system to act as a file and print server to PC clients running the following operation systems:

- Microsoft Windows NT
- Microsoft Windows 95
- Microsoft Windows for Workgroups

Introduction to Samba

Samba is an application of choice allowing interoperability between Windows and UNIX-like systems. It is a group of programs that allows a UNIX host to act as a fileserver for DOS and Windows platforms and also provides print services for them. It is freely available under the GNU Public License. Samba allows UNIX-like machines to be integrated into a Windows network without installing any additional software on the Windows machines. Many different platforms run Samba successfully; and there are nearly forty different operating systems which support Samba.

Features of Samba/iX

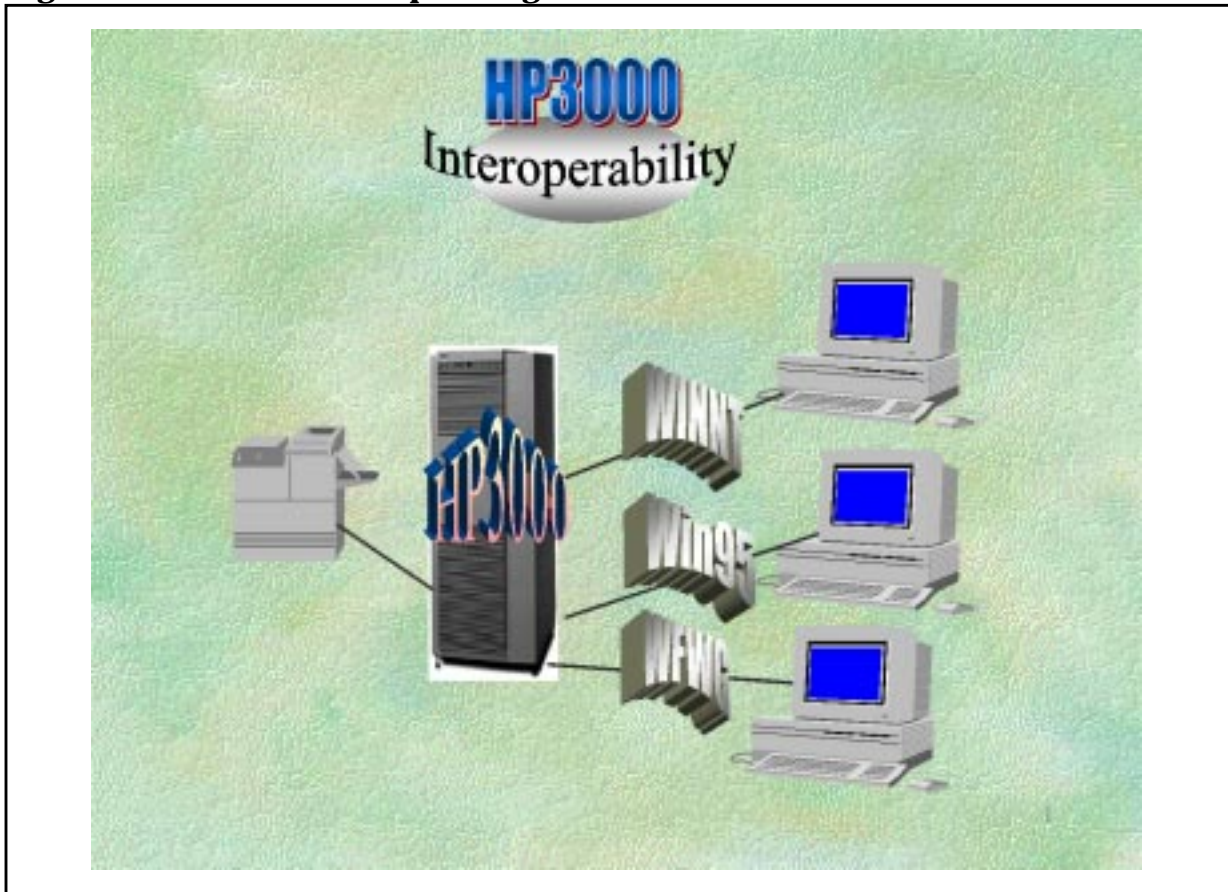
As more of our customers implement and configure networking services in a heterogeneous environment of MPE, UNIX, and Windows NT servers, along with Netware, Windows, and NT workstation clients, the need for knowledge in the area of interoperability becomes a must for our customers. Beginning with MPE/iX release 6.0, Samba/iX is available on MPE/iX shell operating system. It allows clients to access a server's filespace and printers via the SMB protocol.

Samba/iX is the result of porting Samba to MPE/iX under POSIX environment. It is a solution for those wishing to access HP 3000 disk storage and printers (both networked and spooled from MPE/iX) from common PC client operating systems like Windows 95 and NT Workstation.

Samba/iX allows access to these disk and printer resources of MPE/iX, by providing standard SMB file and printer services that are accessible from PC clients and their applications. It is available to the HP 3000 users with the MPE/iX 6.0 release.

A general UNIX program that is part of the Samba suite has also been ported to MPE/iX shell operating system. This program allows MPE users to use an FTP-like interface to access filespace and printers on any other SMB (Server Message Block) servers. This capability enables these operating systems to act like a LAN server or Windows NT server. See Figure 7-1. for HP 3000 interoperating with the Microsoft platforms.

Figure 7-1. HP 3000 Interoperating With Microsoft Platforms



Benefits of Using Samba/iX

There are many benefits in having an MPE/iX and Samba/iX environment, some of which are listed here:

- The remote MPE/iX based POSIX filesystem can be browsed as shared/services from PC clients.
- Remote files can be operated on as if they are stored locally.
- Samba/iX acts as translator between the different file systems for file names and attributes and provides security based on user authentication.
- Samba/iX can support the use of long file names by Windows 95 and Windows NT workstation PC clients.
- Samba/iX provides seamless interoperability between common desktop operating systems, popular PC applications, and HP 3000 through Microsoft network.

Major Components of Samba/iX

Table 7-1., "Major Components," on page 76 shows the major components of the Samba/iX

suite.

Table 7-1. Major Components

SMBD	The SMB server handles connections from clients, performing all the file, permission, and username authentication.
NMBD	The NetBIOS name server advertises Samba/iX on the network, and helps clients locate servers.
SMBCLIENT	Client program on MPE/iX host.
SMB.CONF	Samba/iX runtime configuration file.
TESTPARM	A program to test the Samba/iX configuration file.
TESTPRNS	A program to test server access to printers.

The Samba/iX product contains:

- **SMBD:** This is the server that can provide most SMB services.

The SMB protocol section in the Samba/iX configuration file "SMB.CONF", describes the role of SMB. The HP 3000 running SMBD will act as a File and Print server for the clients using the SMB protocol. This is compatible with the LanManager protocol, and can service LanManager clients.

These clients include Windows for Workgroups, Windows 95 and Windows NT.

A session is created whenever a client requests one. Each client gets a child process for each session. This copy then services all connections made by the client during that session. When all connections from its client are closed, the copy of the server for that client terminates.

- **NMBD:** This is a server that understands and can reply to NetBIOS Name Service Requests on TCP port 137, like those sent by LanManager clients.

NMBD also controls browsing (viewing the resources available on a Windows network is called browsing). When they start up, LanManager compatible clients such as Windows 95/Windows NT, may wish to locate a LanManager server. That is, they wish to know what IP address a specified host is using.

This program simply listens for such requests, and if its own name is specified, it will respond with the IP address of the host on which it is running. Its "own name" is by default the name of the host on which it is running.

- **SMBCLIENT:** The SMBCLIENT is a client that can "talk" to an SMB server.

When this program is run on the HP 3000, it will be acting as a client. It is a command line program and offers an interface similar to that of the FTP program. Operations include things like "getting" files from the server to the local machine, "putting" files from the local machine to the server, retrieving directory information from the server, etc.

- **SMB.CONF:** The SMB.CONF file is a configuration file of the Samba/iX suite which contains runtime configuration information for both SMBD and NMBD.

This file consists of sections and parameters. Each section in the configuration file

corresponds to a service. The special sections are **[global]**, **[homes]** and **[printers]**. The **[global]** section is used to set global configuration options that apply to the server as a whole. The **[homes]** section is designed to grant access to all users home directories and the entries in **[printers]** section correspond to the print services of the Samba/iX server.

- **TESTPARGM:** This is a test program to validate the contents of the `SMB.CONF` configuration file.

If this program reports no problems, you can use the configuration file with confidence that SMBD will successfully load the configuration file.

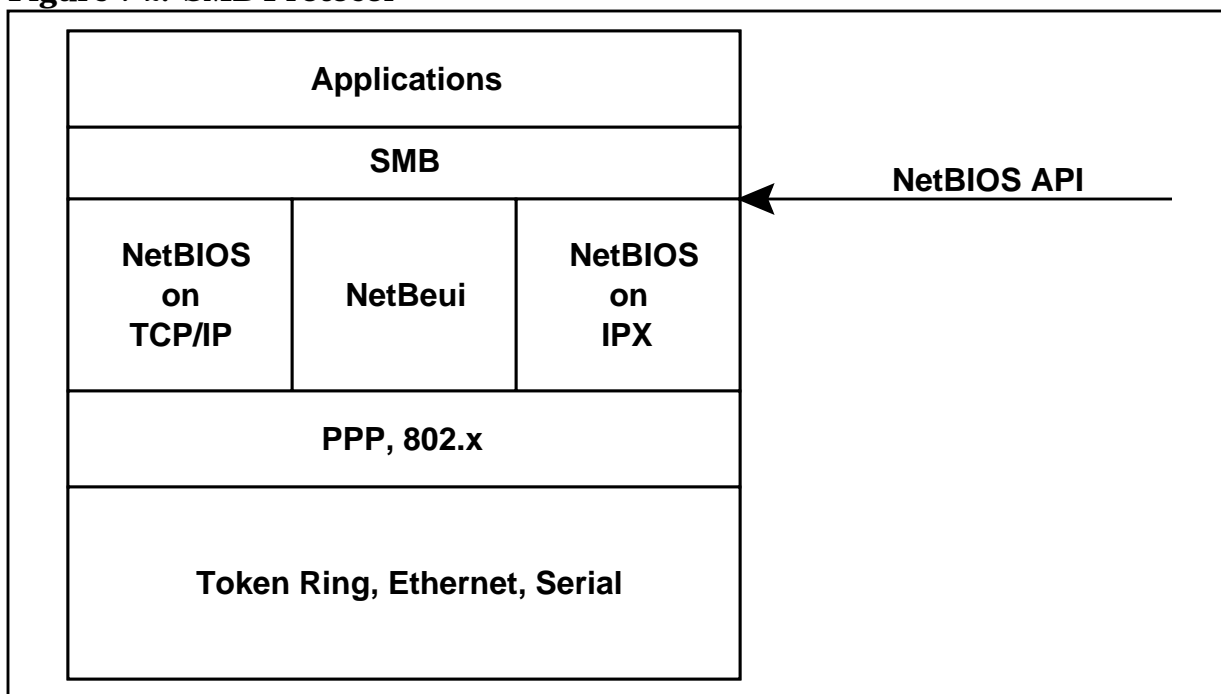
- **TESTPRNS:** This tool checks whether the printer name is valid for the services provided by SMBD.

SMB Protocol

SMB, which stands for Server Message Block, is a protocol for sharing files, printers, serial ports, and communication abstractions, such as named pipes and mail slots, between computers.

SMB is a request/response protocol and it is implemented on top of the NetBIOS API, see Figure 7-2.. It plays the role of session, presentation, and a part of application layer of the OSI stack. SMB can be used over TCP/IP, NetBEUI, and IPX/SPX. In the case of TCP/IP or NetBEUI, the NetBIOS API is being used. Samba/iX uses SMB over TCP/IP.

Figure 7-2. SMB Protocol



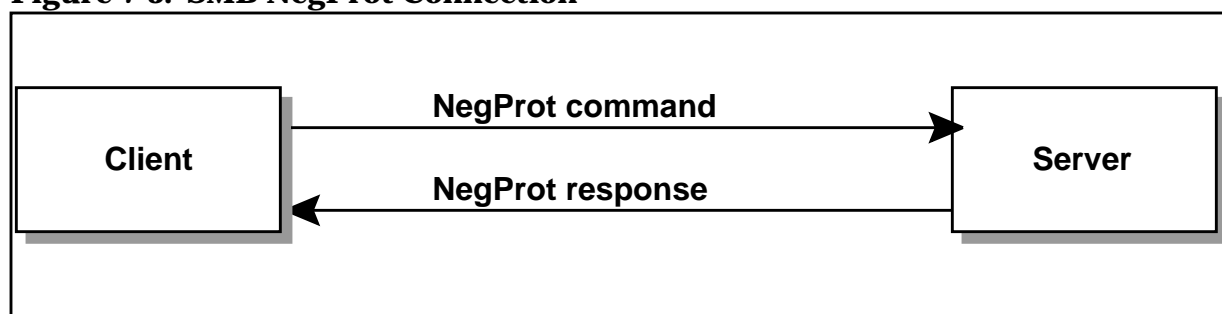
The SMB messages can be categorized into four types of messages: session control, file, printer, and message. Session control messages start, authenticate, and terminate sessions. File command controls file access and printer command controls printer access.

Message commands allow an application to send messages to or receive messages from another host. (For example, WinPopup messages). NetBIOS names are up to 15 characters long, and are usually the name of the computer that is running NetBIOS.

Example of SMB Conversation

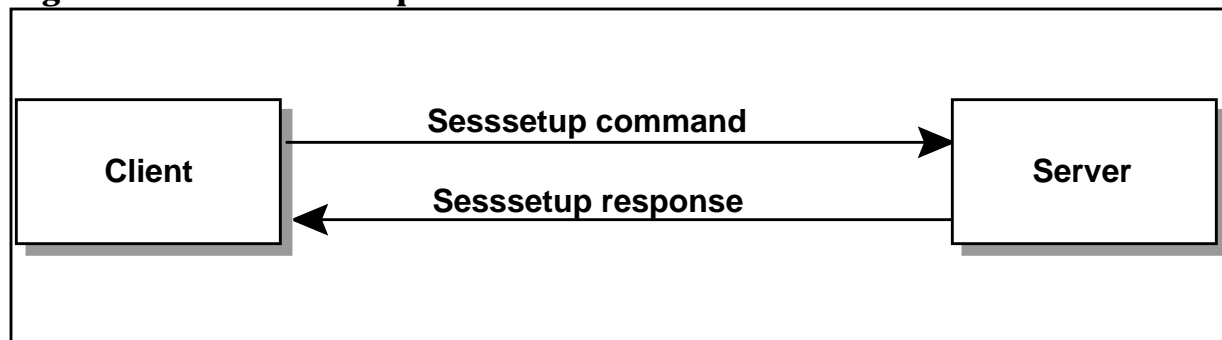
Figure 7-3. demonstrates the process of connecting to a file space service. The SMB Negotiat Protocol command (*NegProt*) is used to decide on a protocol extension to be used with the server. The client sends a SMB *NegProt* to the server. This will list the protocol dialects/protocol extensions that it understands. The server responds with the index of the dialect that it wants to use, or 0xFFFF if none of the dialects were acceptable. Dialects newer than the Core and CorePlus protocols supply information in the *NegProt* response to indicate their capabilities such as max buffer size. The six important protocol extensions of SMB are Core, CorePlus, LAN Manager 1.0, LM 2.0, and NT LM 0.12 and CIFS 1.0.

Figure 7-3. SMB NegProt Connection



Once a protocol has been established, the client can proceed to logon to the server. Client now sends a SMB Session Setup command (*SesssetupX*), see Figure 7-4.. The response indicates whether the username password pair is valid, and if so, can provide additional information. One of the very important aspects of the response is a User ID value that must be submitted with all the subsequent SMBs sent to the server. This is used for user authentication.

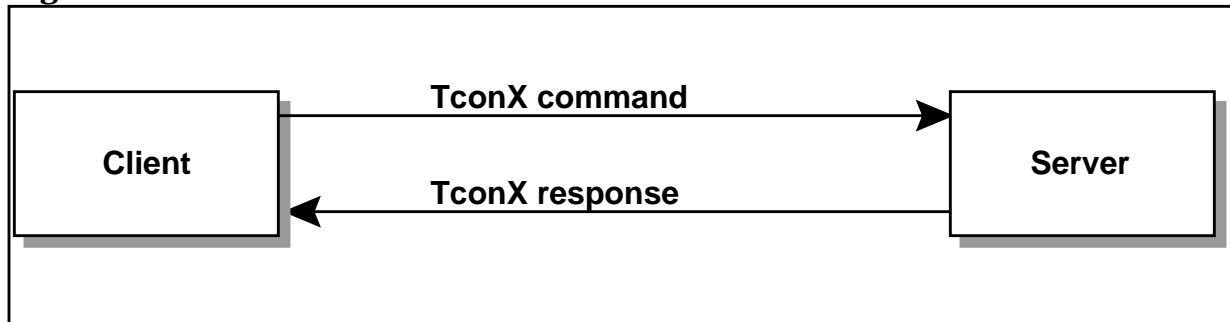
Figure 7-4. SMB Sesssetup Connection



After the client has logged in, then proceeds to connect to the file tree by sending a SMB Tree Connect command (*TconX*) to the server, see Figure 7-5.. Here TconX stands for tree

connect. The client sends a Tcon or SMB TconX specifying the network name of the share that they wish to connect to, and if all is well, the server responds with a TID that the client will use in all future SMBs relating to that share.

Figure 7-5. SMB TconX Connection



After connecting to a tree, the client can now open a file with an open SMB, followed by reading it with read SMBs, writing it with write SMBs, and closing it with close SMBs.

Samba/iX Configuration File Options

The Samba/iX configuration file contains the runtime configuration information for Samba/iX. This file contains the sections and parameters. There are four special sections: the **[global]** section, the **[printers]** section, **[homes]** section and other sections. This file also contains the information required for each share (service) and defines attributes like associated directory path, read or write access for each share.

The Samba/iX configuration file is named "smb.conf" which resides in the /usr/local/samba/lib directory on HP 3000 system. This chapter documents the possible configuration options that the users can specify in the "smb.conf" file. There are many configuration options available, but only the configuration options and uses defined in this manual are supported by HP.

[Global] Section This section is for parameters which apply to the server as a whole rather than to a specific service. It can also be used to specify default values for service-specific parameters which are then inherited by other services, referred to later in the configuration file.

[Printers] Section This section works in conjunction with the printcap file and allows it to configure a large number of printer shares without having to add separate detailed sections for each of them. The printer names and optional aliases are listed in the printcap file; and the configuration parameters are defined in this section.

[Homes] Section This section provides access to the user's home directories without having to add a separate section for each of them. The share name is considered to be a valid user id and the path defaults to that user's home directory.

Other Sections These sections explicitly define the file and printer shares.

Global Configuration Options

The global configuration options can be defined in the **[Global]** Section in the "smb.conf" file, options cover the following configuration options which are supported for use by HP:

- Configuration file option
- Browser option
- Network interface configuration
- Mapping PC usernames to MPE usernames
- Setting the maximum SMB packet size
- Disconnecting idle clients
- Setting logging behaviors
- Login/logout commands
- Global printer service option

Configuration File Option

`config file` The `config file` parameter allows you to specify the pathname for the configuration file used by Samba/iX.

Example: `config file = /usr/local/samba/lib/smb.conf`

Browser Option

`workgroup` The `workgroup` parameter specifies the name of the workgroup; the Samba/iX server will appear as part of the browse list.

Example: `workgroup = SambaIX`

`server string` The `server string` parameter defines the server's comment string. This comment string will appear next to the machine name in the browse lists, such as the network neighborhood.

Example: `server string = HP3000, File/Printer server`

Default: `server string = samba 1.9.16p9`

`default service` This parameter specifies the name of a service that a client will be connected to, if the service actually requested doesn't exist. Typically the default service is some sort of public, read-only service.

Example: `default service = public`

Default: `none`

Mapping PC Usernames to MPE Usernames

`username map` This `username map` parameter allows you to map PC style usernames to MPE style usernames. You can specify the location of your username map file with the `username map` parameters.

Example: `username map = /usr/location/samba/lib/user.map`

The syntax of the username map file is simple. Each line consists of a MPE style name like `manager.sys` and a list of possible PC style username like `webuser`, separated by an equal sign. A sample username map in the `user.map` file is defined as follows.

Example: `manager.sys = webuser`

Network Interface Configuration

`interfaces` The `interfaces` option allows you to inform Samba/iX of each interface you wish it to provide services on, by supplying IP address and subnet mask of your HP 3000 system.

Example: `interfaces = 192.1.2.3/255.255.0`

Setting the Maximum SMB Packet Size

`max xmit` The `max xmit` parameter allows you to set the maximum packet size which Samba/iX can negotiate with a client. This is the maximum packet size that SMBD will accept from a client, setting an upper limit on the packet size that will be negotiated with a client at session setup.

Example: `max xmit = 8000`

Default: `max xmit = 65535`

Disconnecting Idle Clients Option

`dead time` An inactive client will consume server resources even though it is not doing anything. The `dead time` parameter defines an integer value describing the number of minutes of inactivity before a session is automatically disconnected. The “dead time” is considered to begin when a client has no open files. The default “dead time” of zero indicates that no client should ever be dropped because of inactivity.

Example: 5 (in minutes)

Default: 0 (in minutes)

Setting Logging Behavior

`max log size` The `max log size` option specifies the maximum size in kilobytes to which log files can grow. The default value of the maximum log file size is 5000 in kilobytes. If the file exceeds the specified size, it is renamed by adding the `.old` extension.

Example: `max log size = 10000` (in kilobytes)

Default: 5000 (in kilobytes)

`log file` The `log file` parameter allows you to specify the pathname of log file used by SMBD and NMBD processes.

Example: `log file = /usr/local/samba/var/log.smb`

`debug level` The `debug level` parameter allows the debug logging level to be specified in the Samba/iX configuration file. This option defines the level of trace

messages that you want to log into the logfile.

The typical range of the debug level can be from 0 to 5. Large values cause more detailed information to be logged. Most of these debug levels exist to help users to debug the server activity.

Example: `debug level = 3`

Default: `debug level = 0`

Login/Logout Commands

- `preexec` The `preexec` parameter allows you to specify a command to be run whenever the service is connected.
- Example: `callci /usr/local/samba/lib/tellog tcon %S %u %m %I`
- Generates the following example output to the console: 9:41
#J36/50/FROM/MGR.SAMBA/tcon on IPC\$ by MGR.SAMBA from rkm-nt
- `postexec` The `postexec` parameter allows you to specify a command to be run whenever the service is disconnected.
- Example: `callci /usr/local/samba/lib/tellog tdis %S %u %m %I`
- Generates the following example output to the console: 9:41
#J36/70/FROM/MGR.SAMBA/tdis on IPC\$by MGR.SAMBA from rkm-nt

Global Printer Service Options

The global printer service options allows you to specify the location of the “`printcap`”, printer command parameter used by Samba/iX.

The following global printer configuration options are supported for use by HP:

- `load printers` The `load printers` parameter is used in conjunction with `printcap` file and `[printers]` section. It is a boolean variable that controls whether all printers in the “`printcap`” file will be loaded for browsing.
- If the `load printers` parameter is set to true, all printers defined in the `printcap` file will be loaded for browsing by default.
- Example: `load printers = yes`
- Default: `load printer = no`
- `printcap name` The `printcap name` option specifies the location of the `printcap`. Samba/iX uses the `printcap` to determine all printers available on the system if the general `[printers]` service is used instead of defining each printer in its own service.
- Example: `printcap name = /usr/local/samba/lib/printcap`
- `print command` The `print command` parameter defines the shell command which Samba/iX will use to submit a print job. After Samba/iX has finished spooling a print job to the disk, it calls this command. After processing the file, this command must remove the spoolfile, unless you don't mind spool files building up on your system.

This parameter can use the following print-specific macros:

`%s` The full path of the print spool file.
`%p` The name of the printer to which the job is to be submitted.

Example: `print command = /usr/local/samba/lib/rawlp %s %p; rm %s`

On MPE/iX 6.0, the “rawlp” utility is available on the system and is used to send the file contents to a spooler like “lp -oraw”.

Controlling User Access Rights

`allow hosts` Default: none

`deny hosts` These parameters allow users to define a set of client IP addresses which will be granted access to service. If an allow hosts option is present, only hosts matching the pattern are allowed to access the service. If a deny hosts option exists, only hosts not matching the pattern will be granted access.

Example: `allow hosts = 192.1.2.3`

Default: none

`valid users` Default: none

`invalid users` If neither of these parameters are set, then any authenticated user will be granted access to the service. The valid users parameter may contain a comma-delimited list of users who will be allowed to access the service. The invalid users parameter may contain a similar comma-delimited list of users who will never be granted access to the service. These parameters use MPE style user syntax (for example, `user.acct`) to specify users. The password format used when you log on from a PC client should be `userpassword, acctpassword`.

Example: `valid users = mgr.samba`

Default: none

`guest account` The shares can be configured to accept connections without a validated user ID and password, then you can use the “guest account” parameter to assume the guest logon identify for accessing files and printers.

Example: `guest account = mgr.samba`

Default: none

`revalidate` Forces the revalidation of password. When Samba/iX successfully validates a clients password, it passes a token back to client. This is used by the client to connect to other shares. If `revalidate=true`, then Samba/iX expects a valid username and password pair again without just relying on the token. For example, after connecting to “temp”, if the client tries to connect to another share, Samba/iX revalidates the password.

Example: `revalidate = yes`

Default: no

Share Configuration Options

This section covers the share configuration options that you use when you configure for a specific disk or printer share in the Samba/iX configuration file.

Setting the Shared Directory

`path` The path parameter specifies the pathname of the shared directory.

Example: `path = /usr/local/samba/docs`

For printer services, this parameter describes the directory used to temporarily spool files sent from clients for printing before they are spooled to the local HP 3000 printer.

Example: `path = /usr/local/samba/spool`

Browser Option

`browseable` This parameter controls whether this share is seen in the list of available shares in the browse list.

Example: `browseable = yes`

Default: `browseable = yes`

`Available` This parameter lets you remove a service from availability. If available is no, all attempts to connect to the service will fail. Using this option preserves the service's settings and is usually more convenient than commenting out the service.

Example: `available = no`

Default: `available = yes`

Comment Option

`comment` The "comment" parameter specifies the comment message in the share services.

Example: `comment = share "public" service for guest users.`

Printing Access

`print ok` The "print ok" option is specified in the `[prints]` section to enable the share for printing access.

Controlling Read/Write Access

`guest ok` If `guest ok` is true, then guest access will be allowed. The access rights of a client connecting as guest will be those of the username set in the "guest account."

Example: `guest ok = yes`

Default: `guest ok = no`

- `guest only` If `guest only` is true, then access of service/share is only granted with the rights of usernames given in the “guest account” parameter.
Example: `guest only = yes`
Default: `guest only = no`
- `create mode` The “create mode” is used to define the permission used by share services. This option sets an octal value representing the file permissions available to a file created by Samba/iX.
Example: `create mode = 0744`
The value of 0744 causes the group and other write and execute bit to be removed from a file created by Samba.
- `read only` Example: `read only = yes`
Default: `read only = yes`
- `write ok` The `read only = yes` is identical to `write ok = no`. If `write ok` is true, clients will be granted read/write access to a share. The same effect can be achieved by setting `read only` to false.
Example: `write ok = no`
Default: `write ok = no`

Sample Configuration File — `samp-smb.conf`

When you want to use Samba/iX, you should copy the Samba/iX sample configuration file to `/usr/local/samba/lib/smb.conf` and adjust this file as needed. The sample configuration file `samp-smb.conf` resides in the `/usr/local/samba/lib` directory. Please refer to Appendix A , “Samba/iX Sample Configuration File.”

Configuring the Shares for File Sharing

The PCs can access the server side filesystems using Samba/iX. Whenever the clients want to connect to the server, the server side validates the username and password which are sent by the client and grants access to the requests share if it is appropriate.

You can configure the file service with guest access and the Samba/iX server can grant to the guest users without a validated user ID and password.

Share level security is the default security level in Samba/iX. The following example shows the configuration steps you can use to configure with `[global]` and `[service]` section with `security = share`:

1. Add in the `[global]` section the following parameter: `security = share`
2. To add a share, the entries can be given in the example below:

`[sample shares]`

```
comment = shared space
guest ok = no
write ok = yes
```

```
path = /sample/test
```

3. Add a username mapping in “user.map” file. For example: mgr.sample = pcusername
4. When you connect a share from a PC, the password format that you enter from a PC should be userpassword, acctpassword.

NOTE For accessing share/user security modes, both SAMBA account and MGR.SAMBA user should have PM capabilities.

Configuring a Printer Section for Printer Sharing

The PCs can access the server side printer using Samba/iX. With printer sharing the client creates a file on the server directory associated with the printer, and then lets the server process trigger a configurable command to push the file into the MPE spooler.

The [printers] section works in conjunction with the printcap file and allows to configure a large number of printer shares without having to add separate detailed sections for each of them. The printer names and option aliases are listed in the printcap file.

Here is an sample example of printer names in the samp-printcap file which resides in /usr/local/samba/lib:

samp-printcap file:

```
LP|6|HP3000 System LP
```

Here is a sample example for the configuration option that you may configure with [global] and [printers] sections in the Samba/iX configuration file — smb-conf:

```
[global]
# You need supply IP address and subnet mask of your 3000 with the interface
parameter
interface = ip address/subnet mask
# printcap file lists printer names for use by [printer] section
printcap name = /usr/local/samba/lib/printcap
# shares may be configured to accept connections without a validated user id
and password, and then it assumes the guest logon for accessing the printers.
guest account = mgr.samba
[printers]
# enable this service for printing but not for file access
print ok = yes
write ok = no
# current version of Samba/iX only allows guest users for printer sharing
guest ok = yes
guest only = yes
# the “staging” directory for print requests
path = /user/local/samba/spool
```

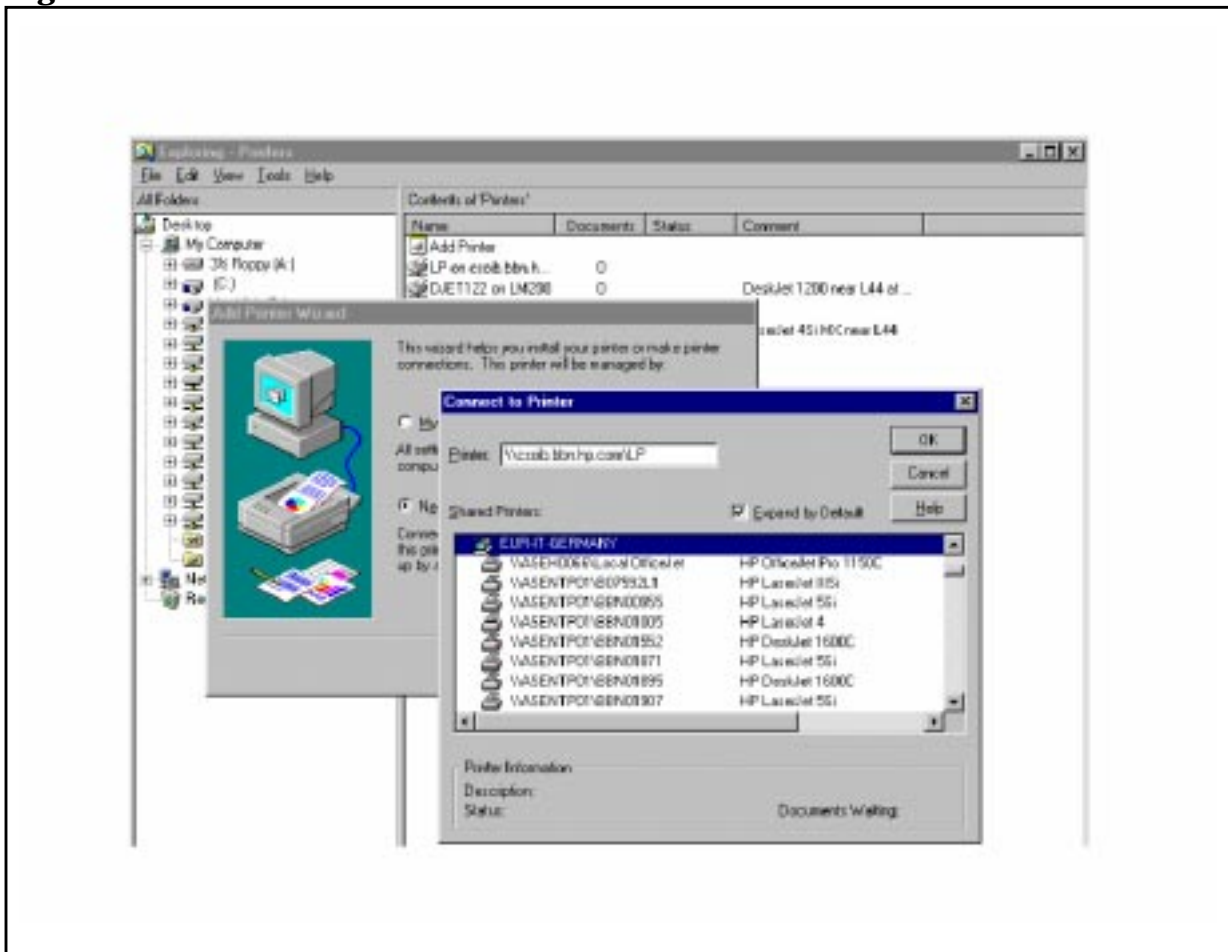
```
# The rawlp utility sends file contents to spooler like "lp -oraw"  
print command = /usr/local/samba/lib/rawlp %s %p; rm %s
```

NOTE For the current version of Samba/iX, the printer sharing only works for guest users.

The current configuration option for printer sharing needs to set “guest ok” and “guest only”.

Add a printer, as shown in Figure 7-6.. With printer sharing, the printers are accessible to HP 3000.

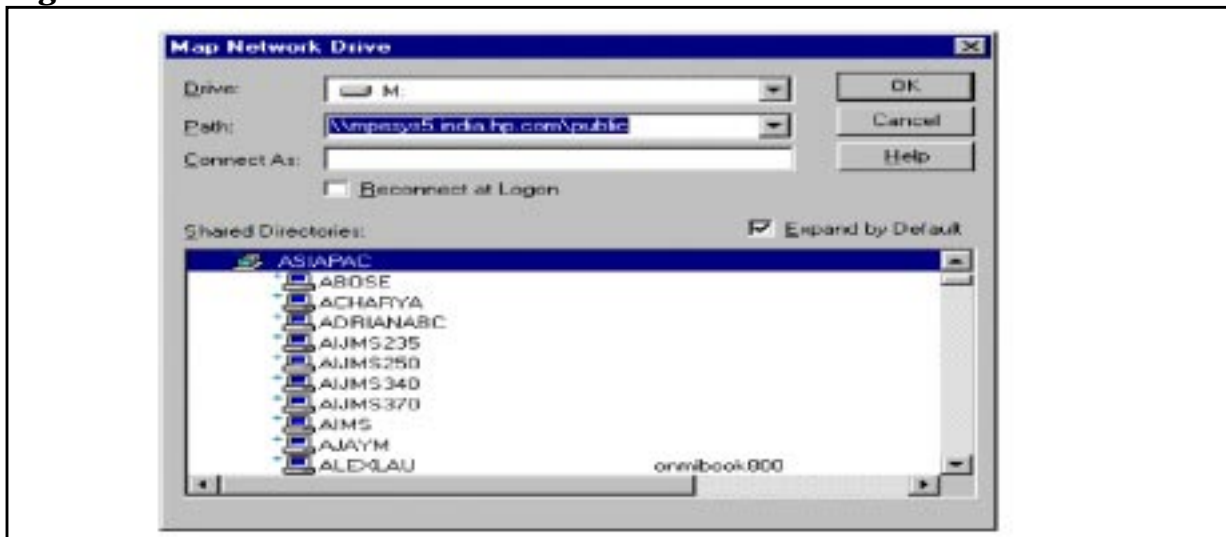
Figure 7-6. ADD a Printer



You can connect your server shares using the NT explorer, as shown in Figure 7-7..

The menu tool includes a “map network drive” which brings up the small windows shown in Figure 7-7.. You connect a network driver by typing in a share name with \\servername\sharename syntax in the “path” box.

Figure 7-7. Connect to the HP 3000 Shares



You can view the contents of the share from NT explorer, as shown in Figure 7-8.. Click the share name at NT explorer window, it will list the files residing in this share

Figure 7-8. View the HP 3000 Share



Starting and Stopping Samba/iX

This section covers the steps to start or stop Samba/iX.

Starting Samba/iX

Before you start to run Samba/iX server or client components, you should have set up the TCP/IP networking on your HP 3000 system as well as your PC. On the HP 3000 system side, you should have a proper IP address and subnet mask configured in NMMGR as well as `NETCONTROL START` successfully executed. You must choose to start `SMDB` and `NMBD` either as listener jobs or under control of `INETD`.

Disable Resource Sharing

If your system has `NBDAEMON.PUB.HPLANMGR` running, then `SMDB` and `NMBD` will not be able to use ports 137 and 139 as `NBDAEMON` already binds to them. The workaround solution is to stop the `PDSERVER` process. The `NBMON` and `NBDAEMON` processes will not start because of this workaround. This can be done by modifying the file `PDSSERV.NET.SYS` by changing the line 7 from 1 to 0. This will setup `PDSSERVE` for non-reserved servers.

The following shows the steps of making non-reserved servers:

1. Modify the file `PDSSERV.NET.SYS` and change the line 7 from 1 to 0.
2. Shutdown the network.
3. Stream `JCONFJOB.NET.SYS`.
4. Start the network back up.
5. The command `nscontrol status=services` should show non-reserved `PDSERVERs`.

Verify Link Configuration

The default assumes that LAN link configuration in NMMGR is `SYSLINK`. You need to run the following command to get the IP address and subnet mask of your HP 3000 system; you will need this information for future Samba/iX configuration file updates with the “interfaces” parameter.

1. Logon as `manager.sys`
2. Enter the command `Netcontrol status; net=LAN1`

The following example displays when you run the command `netcontrol status; net = lan1`.

```
NETWORK NAME:          LAN1
NETWORK IP ADDRESS:    $0F0DC750 15.13.188.80
NETWORK SUBNET MASK:   $0FF000000 255.0.0.0
```

Add PM Capability

To access share security modes, both `samba` and `mgr.samba` user accounts should have PM capabilities.

1. Logon as `manager.sys`
2. Add PM capability to `samba` account

3. Add PM capability to `mgr.samba` user

Starting SMBD and NMBD Listener Jobs

1. Logon as `mgr.samba`
2. Copy the sample configuration file `samp-smb.conf`, `samp-printcap` and `samp-user.map` to `smb.conf`, `printcap` and `user.map`. Modify the entries to suit your Samba/iX environment. The `samp-smb.conf`, `samp-princap` and `samp-user.map` files reside in the `/usr/local/samba/lib` directory.
3. Check your Samba/iX configuration files with `TESTPARG` utility. The `TESTPARG` utility resides in the `/usr/local/samba/bin` directory. Run the following command:
`shell/ix> testparm /usr/local/samba/lib/smb.conf.`
4. Start your SMBD listener and NMBD server.
5. Use the jobs supplied as `JSMB.SAMBA.SYS` and `JNMB.SAMBA.SYS` and stream them.
6. Use `SHOWJOB` to see if the jobs stay alive, it can look as follows:

JOBNUM	STATE	JIN	JLIST	JOB NAME
#J30	EXEC	10S	LP	NMBMON, MGR.SAMBA
#J31	EXEC	10S	LP	SMBMON, MGR.SAMBA

Starting Samba/iX Under the INETD Control

If you choose to run SMBD and NMBD processes under control of INETD, you should have new entries in `SERVICES.NET.SYS` and `INETDCNF.NET.SYS` and then you have to create symbolic links to make `SERVICES.NET.SYS` link to `/etc/services` and `INETDCNF.NET.SYS` symbolic links to `/etc/inetd.conf` respectively. Perform the following steps:

1. Logon as `manager.sys`.
2. Copy `SERVSAMP.NET.SYS` file to `SERVICES.NET.SYS` if `SERVICES.NET.SYS` doesn't exist. The following two entries should exist in file `SERVICES.NET.SYS`:

```
nmbp 137/udp
smbp 139/tcp
```

3. Copy `INCNFSMP.NET.SYS` file to `INETDCNF.NET.SYS` if `INETDCNF.NET.SYS` doesn't exist, the following two entries should exist in file `INETDCNF.NET.SYS`:

```
nmbp    dgram    udp    wait    MGR.SAMBA    /SYS/SAMBA/NMBD    nmbd
smbp    stream    tcp    nowait  MGR.SAMBA    /SYS/SAMBA/SMBD    smbd
```

4. Use the following two commands to create symbolic links to make `SERVICES.NET.SYS` link to `/etc/services` and `INETDCNF.NET.SYS` links to `/etc/inetd.conf`, respectively:

```
:newlink /etc/services, /SYS/NET/SERVICES
:newlink /etc/inetd.conf, /SYS/NET/INETDCNF
```

5. Stream `JINETD.NET.SYS` to start SMBD listener and NMBD server (or use `INETD -c` to reread configuration file if INETD is already running.)

6. Use `SHOWOUT JOB= Jobnumber`
7. Print `Oxxx.OUT.HPSPOOL` to check for any problems in the spool files.

In case of problems, check for the job listings for useful error messages and look into the Samba/iX log file `/usr/local/samba/var/log.smb` and `log.nmb` for hints. You can control the amount of log messages with the “debug level” directive inside the config file.

Stopping Samba/iX

It is important to shutdown Samba/iX before bringing the system down. You can use the following commands to stop Samba/iX:

1. Use `SHOWJOB` to see if the jobs stay alive, it can look as follows:

JOBNUM	STATE	JIN	JLIST	JOB NAME
#J30	EXEC	10S	LP	NMBMON, MGR. SAMBA
#J31	EXEC	10S	LP	SMBMON, MGR. SAMBA

2. Use the following two commands to stop Samba/iX:

```
:abortjob          #smbjobnumber
:abortjob          #nmbjobnumber
```

NOTE Clients connected and writing to files will loose data if an `abort job` is done with clients active.

Initial Test With smbclient Utility

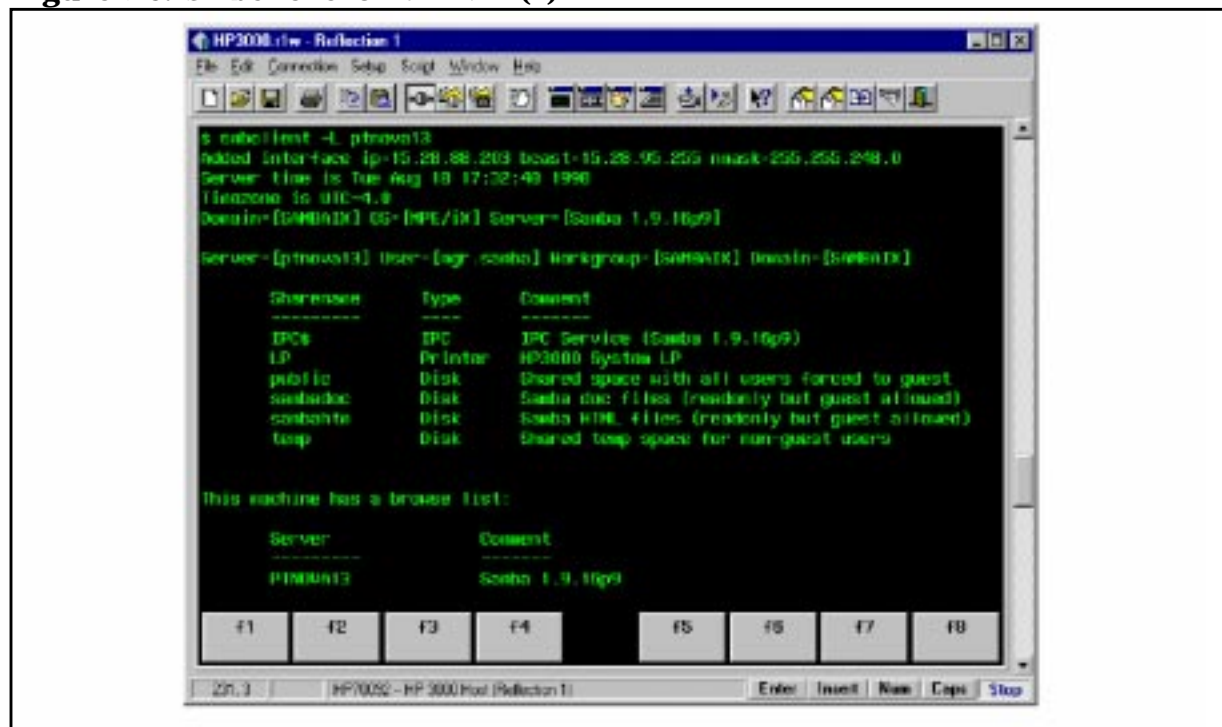
The `smbclient` utility provides access to SMB servers with an FTP-like user interface. You can run `smbclient` utility on POSIX/Shell environment.

Logon to your MPE system as `mgr.samba`:

```
: sh.hpbin.sys
shell/iX> cd bin
shell/iX> smbclient -L <sambaserver>
```

This command should display a list of available shares (services) that matches your configuration file. If NMBD is running, a list of workgroups and related computers that NMBD could find on your network/subnet will be displayed, see Figure 7-9..

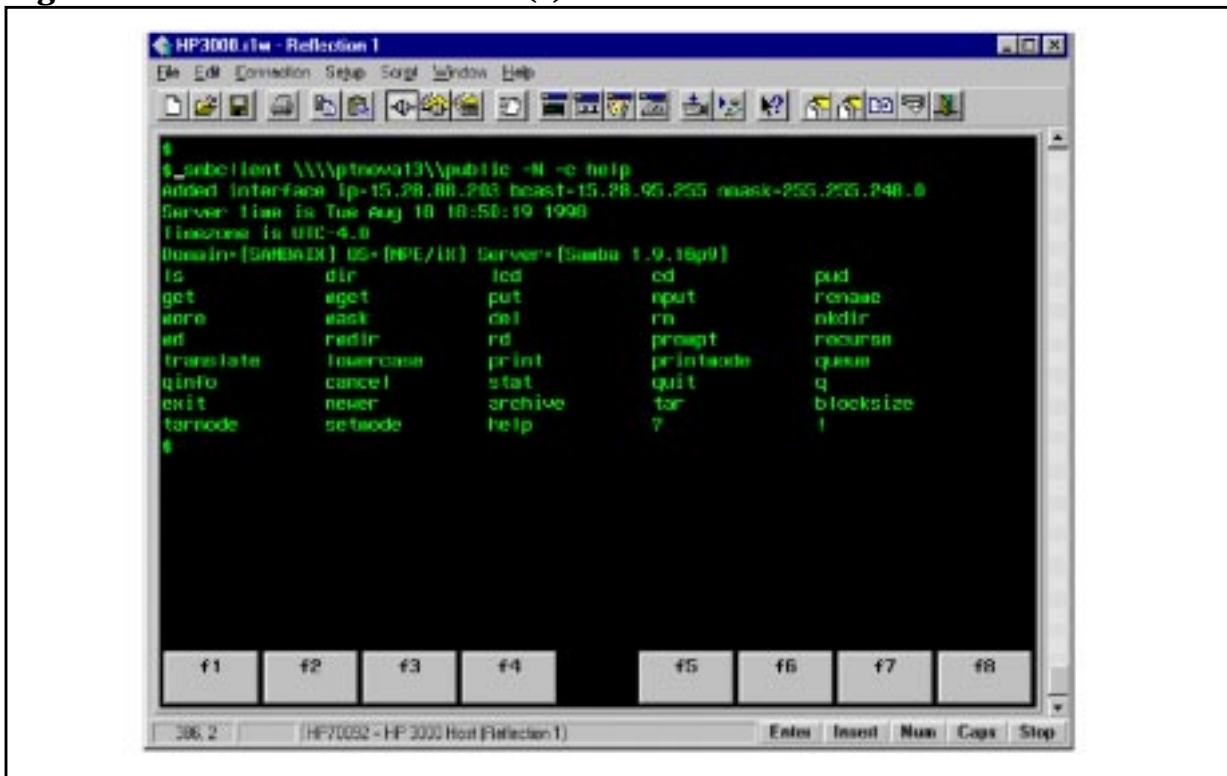
Figure 7-9. smbclient for MPE/iX (1)



```
shell/iX> smbclient \\\<sambaserver>\\sambadoc -N -c help
```

This command should connect to the sambadoc share on your HP 3000 using -N to suppress password prompt and effectively become guest user and display the contents of on-line help screen of smbclient, see Figure 7-10..

Figure 7-10. smbclient for MPE/iX (2)



NOTE All smbclient examples used the -c option to specify the command on the command line. The smbclient program has an interactive mode which looks like FTP. Due to limitations of the select() system call on MPE/iX, the interactive mode does not yet work properly. At present, it can be worked around by using the -c option of smbclient

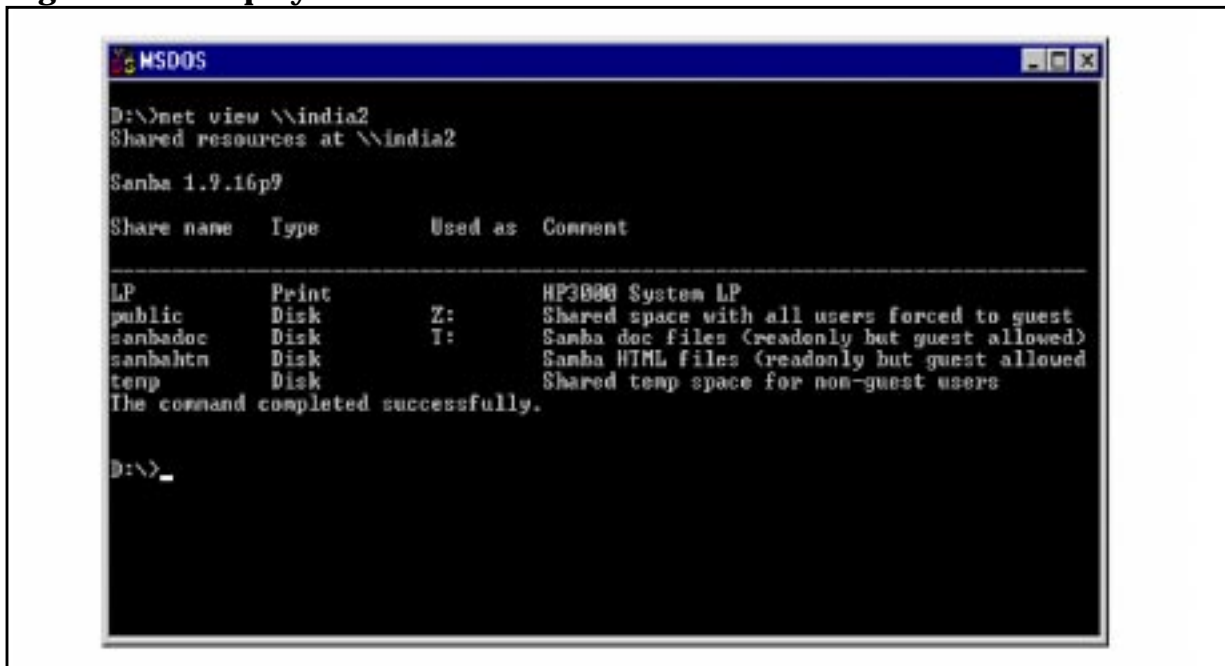
Initial Test From a PC Client at DOS Prompt

You can open a DOS command window and issue the command line using the following commands for initial test from a PC client:

```
C:\> net view\\servername
```

This command, will display a list of available shares for the server, see Figure 7-11..

Figure 7-11. Display Available Shares From a PC Client



If you want to display a list of available shares on the Samba/iX server named “HP 3000; enter the following command at the DOS prompt:

Example: C:\> net view \\hp3000

C:\>net use x:\\servername\servicename

This command will connect to a network drive X by entering the sharename \\servername\servicename.

If you want to connect to drive letter “X” from your PC to the “Sambdoc” service on Samba/iX server named “HP 3000”, type the following command at the DOS prompt:

Example: C:\> net use x: \\hp3000\sambdoc

Samba/iX Share level Security Mode

The process of user authentication depends whether Samba/iX is running in share level or user level. The “security” parameter in the configuration file is used to specify the share level or user level authentication. If the “security” parameter is set to share, Samba/iX will tell clients it is granting access under share mode security. The process for granting access under share level security is:

- If the service is marked “guest ok” or “public”, then the client is granted access with the rights of the username given in the “guest account” parameter for the service.
- If a service is marked as “guest only” (not guest ok or public) then access is granted

with the rights of the username given in the guest account parameter for the service.

- If a client passed a username/password pair to Samba/iX, and the username and password are validated, the client is granted access with the rights of the username.
- If the client registered a username with Samba/iX during a previous connection and now supplies the correct password for that username, access is granted.
- If the client validated a username/password pair with the Samba/iX server during a previous connections, and now passes the correct corresponding access token, access is granted. This step will be skipped if the “revalidate” service parameter is true for this service.

Samba/iX Server Security Mode

Samba/iX server mode security is just one of the security policies of user level authentication. This mode of security is one of the types in processing user authentication. After the user is validated, access rights are enforced for the user:

To make Samba/iX operate in server security mode:

- Add `security = server` in the **[global]** section for `smb.conf` specifying `security = server` in `smb.conf`, the server security mode is on.
- Add `password server = <yourNTserver>`

This option will allow Samba/iX to ask a remote SMB server for password checks, for example, a Windows NT server. This option will be useful if you are integrating an MPE/iX into an already existing NT domain. It is better to set your Windows NT (primary or backup domain controller) server as the password server.

Please set the password parameter to the DNS name of the Windows NT server.

NOTE For the server security mode to work, Windows username should be mapped to the user and account on the MPE/iX host. Now the users have to provide correct passwords for the MPE/iX user and account name while logging on to the server.

After setting up the above configuration, client can proceed to login to the Samba/iX server. When connecting to a service using user level security, the client sends a session setup SMB that includes username and password. This step is not necessary while using shared level security.

In server level security, the Samba/iX server reports to the client that it is in user level security. The client sends username and password pair. The Samba/iX server takes the username/password that the client send and attempts to login to the “password server” by sending exactly the same username/password that it got from the client. If that server is in user level security and accepts the password then Samba/iX accepts the clients connection. This allows the Samba/iX server to use another SMB server as the “password server”.

Some particular issues with Samba/iX and Windows NT; one of the problems with Windows NT is that NT refuses to connect to a server that is in user level security mode and that doesn't support password encryption unless it first prompts the user for a password.

This means even if you have the same password on the NT box and the Samba/iX server you will get prompted for a password. Entering the correct password will get you connected.

Troubleshooting Samba/iX Server

This section covers a list of tests you can perform to validate or diagnose your Samba/iX server. If it passes all these tests, then it is probably working fine.

Prerequisites

In all of the tests it is assumed you have a Samba/iX server 1.19.16p9 or later running on your HP 3000. It is also assumed that the PC is running Windows for Workgroups, Windows 95 or Windows NT with a recent copy of the Microsoft TCP/IP stack. All these tests should be done with Windows for Workgroups (WfW), Windows 95, Windows 98 and Windows NT clients, as they all use different SMB's for file operations.

You need to have a sample share called "public" for testing purposes. Check to see if you have "public" share in smb.conf file:

```
[public]
    comment = files are shared
    path = /SAMBA/SHR/public
    read only = yes
```

Troubleshooting Procedures

Please follow these tests for diagnosing your Samba/iX server.

TEST 1:

In the directory in which you store your smb.conf file, run the command `testparm smb.conf`.

If it reports any errors then your smb.conf configuration file is faulty.

TEST 2:

On the client side; open MS-DOS prompt and run "ping SAMBAIXSERVER" from the PC and "ping CLIENTPC" from the HP 3000 system. If you don't get a valid response, then your TCP/IP software is not correctly installed.

If you get a message saying "host not found" or similar then your DNS software or hostname is not correctly set up.

Ping might fail, if your host is running firewall software. You will need to relax the rules to let in the workstation in question, perhaps by allowing access from another subnet.

TEST 3:

Run the command `smbclient -L SAMBAIXSERVER` on the HP 3000 system. You should get a list of available shares back.

If you get a “connection refused” response, then the SMBD server could not be running.

If you get a “session request failed” then the server refused the connection to SMBD. Check your config file (`smb.conf`) for syntax errors with “testparm” as well as the various directories where Samba/iX keeps its log and lock files.

Another common cause of these two errors is having something already running on port 139 (as in the case of NBMON/NBDAEMON) or SMBD already running under INETD.

And yet another possible cause for failure of TEST 3 is when the subnet mask and/or broadcast address settings are incorrect. Please check that the network interface IP Address/Broadcast Address/Subnet Mask settings are correct and that Samba/iX has correctly noted these in the `log.nmb` file.

TEST 4:

Run the command `nmblookup -B SAMBAIXSERVER __SAMBA__` on the HP 3000. You should get the IP address of your Samba/iX server.

If you don't get the IP address, then NMBD is incorrectly installed. Check your INETD, if you run it from there, or that the daemon is running and listening to UDP port 137.

Check your INETD entries related to `nmbd`, as discussed earlier.

TEST 5:

Run the command `nmblookup -B CLIENTPC '*'` on the HP 3000.

You should get the PC's IP address. If you don't get the PC's IP address, then the client software on the PC is not installed correctly, the PC is not started, or you have the name of the PC wrong.

TEST 6:

Run the command `nmblookup -d 2 '*'` on the HP 3000.

This time try the same as the previous test but, try it via a broadcast to the default broadcast address. A number of NetBIOS/TCP/IP hosts on the network should respond, although Samba/iX may not catch all of the responses in the short time it listens. You should see “got a positive name query response” messages from several hosts.

If this doesn't give a similar result to the previous test, then `nmblookup` isn't correctly getting your broadcast address through its automatic mechanism. In this case you should experiment using the “interfaces” option in `smb.conf` to manually configure your IP address, broadcast and netmask.

If your PC, and server aren't on the same subnet, then you will need to use the `-B` option to set the broadcast address to that of the PC's subnet.

This test will probably fail if your subnet mask and broadcast address are not correct. (Refer to TEST 3 notes).

TEST 7:

On the PC, type the command `net view \\SAMBAlXSERVER`. You will need to do this from within a “DOS prompt” window. You should get a list of available shares on the server.

If you get a “network name not found” or similar error then NetBIOS name resolution is not working. This is usually caused by a problem in NMBD. To overcome the error you could do one of the following (you only need to choose one):

- Fix the NMBD installation.
- Add the IP address of SAMBAlXSERVER to the “wins server” system in the advanced TCP/IP setup on the PC.
- Enable Windows name resolution via DNS in the advanced section of the TCP/IP setup.
- Add SAMBAlXSERVER to your `lmhosts` file on the PC.

TEST 8:

Run the command `net use x: \\SAMBAlXSERVER\Public`. You should get a “command completed successfully” message. If not, your PC software is incorrectly installed or your `smb.conf` is incorrect.

TEST 9:

Run the following command to test the print services.

- `smbclient '\\sambaserver\lp' -P -c “print testfile”`

If printing itself is a problem check the `/usr/local/samba/lib/printcap` file. Format of the file is simple.

```
printername | printer description
```

Printername must equal one of the printer names you normally print to using MPE/iX. The description can be any free text.

```
LP|HP Laserjet in printing room
```

On the PC:

- `net use lpt1: \\sambaserver\lp as guest`

Print test page/pages to the printer connected to the Samba/iX server. At the command prompt type `copy test.txt \\sambaixserver\lp`.

TEST 10:

Some other tests, along with the ones mentioned previously, might be useful. These tests can be done to check the behavior of the Samba/iX server with these security policies:

1. Configure Samba/iX in User security mode:

- map a PC username to a valid MPE/iX `username.account` with passwords
- verify file and print access work
- verify files created by PC user are owned by correct MPE/iX username and account
- verify full file read and create access to the user’s default home share.

2. Configure Samba/iX in Share security mode, and set passwords on file shares.
 - verify that the file and print access from PC users works.
3. Configure Samba/iX in Server security mode, pointing user validation to a NT server.
 - verify users logged into the Windows NT domain being used as a validation server have the appropriate access to shares and printing on Samba/iX.
4. Perform PC connectivity and file/print access tests with SMBD and NMBD in daemon mode (for example, started from MPE/iX jobs JSMB and JNMB) as well as started from INETD as services.
5. Verify all functionality works when the daemons or services are running as the default `mgr.samba`. If any functionality does not work, check if any change needs to be done in the default capabilities of `mgr.samba`.

Using Logfiles of Samba/iX

In case of problems, check for the job listings for useful error messages and also look into the Samba/iX log file `/usr/local/samba/var/log.smb` and `log.nmb` for hints. You can control amount of log messages with the “debug level” directive inside the config file `smb.conf`.

Increasing the log level to 3 or 4 can shed light on the cause of most problems. This also may lead to a large amount of details to be logged into these files.

You may have to increase the size of your log file if your debug level is more than 3.

NOTE Before using the logging feature of Samba/iX, make sure you check the Electronic Support Center (ESC) for information on any possible Samba/iX problems. (use of the HP Electronic Support Center is governed by the HP Electronic Support Center Terms and Conditions.) URL:
<http://us-support.external.hp.com/>

8 DNS BIND/iX

BIND (Berkeley Internet Name Domain) is an implementation of the Domain Name System (DNS). It consists of a network of servers which provide a distributed database, including names and addresses of host machines. This information is accessible to client hosts which are running resolver software. This enables them to send queries to and receive replies from the servers.

The resolver software runs on MPE/iX versions preceding 6.0 so that the MPE/iX client hosts can query DNS servers running on other platforms. On MPE/iX 6.0 there is a full implementation of BIND which means that your MPE/iX host can now act as a DNS server on your network.

Introduction

This section of the Configuring and Managing MPE/iX Internet Services manual assumes that the reader has prior experience with DNS BIND as implemented on other operating systems, or has familiarity with the concepts involved. There are a number of good textbooks available on this subject to which the reader is recommended — the following is a brief overview of a sophisticated system.

The Domain Name System is a distributed and structured directory of information. One of its more frequent uses is the naming of host machines. A DNS host name will consist of several fields separated by dots, for example:

```
quasar.india.hp.com.
```

The host `quasar` exists in the domain `india`, which itself is a subdomain of `hp`, which is a subdomain of `com`, which is a subdomain of the root domain (identified as “.”).

With this structured naming convention, the responsibility for maintaining accurate database information for a name domain can be delegated to a server which is managed by the organization who owns that domain. For example, DNS server hosts within HP maintain information about `hp.com`. Queries for names inside the domain `hp.com` will be referred to that server by servers in other domains. Within HP, the responsibility for `india.hp.com` can also be delegated to another local DNS server.

Before MPE/iX 6.0, hosts running MPE/iX were able to make DNS queries of servers running on other machines and operating systems. Now a full implementation of the server code has been introduced. DNS BIND/iX will enable your MPE/iX host to act as a DNS server, both responding to queries (from clients and other servers) as well as communicating with other DNS servers on the local network and the Internet.

The way this information is accessed is through client programs or code routines called “resolvers”. When a program on a client host needs to obtain information about a domain, it will send a message to the local DNS server host. If the local server has this information, it will send back a reply immediately. If the local server does not have this information, it will research by sending queries to other servers, following the Domain Name System structure. Once the local server has found an answer for the client, it will then reply, but will also cache what it has learned in order to respond more speedily to subsequent queries.

DNS BIND/iX on MPE/iX 6.0 is an implementation of BIND version 8.1.1, which has introduced many new features since the more commonly used version 4.9.4, (with which the majority of experienced DNS users will be familiar).

This is the latest version of BIND, 8.1.1. with features like:

- DNS Dynamic updates
- DNS change notification
- completely new configuration syntax
- flexible and categorized logging system
- more efficient zone transfers

The package contains a host of utilities and administration tools:

- `nslookup` — query Internet name servers interactively
- `dig` — Domain Information Groper
- `host` — look up host names using domain server
- `addr` — get address of host
- `dnsquery` — give all the DNS details and Mail exchange records

Explanation of Terms

BIND, which stands for Berkeley Internet Name Domain, is the most commonly used implementation of DNS.

DNS is essentially a distributed data base, with control of the different elements of the data base maintained by individuals responsible for the domain served by that DNS server. The data is used by DNS servers to assist one host in identifying the location of another host anywhere in the system, translating a host name to its IP address, and visa versa.

The DNS distributed data base is much like a directory. It is organized in an inverted tree fashion, much like the unix directory structure, with the most inclusive node, or domain, at the top, with multiple levels of sub-domain names below, until at the end are the actual host names.

Information about each domain, specifying the sub-domains or hosts below it, are maintained in the DNS data base files. The convention is to call these files “db files” in BIND 4.X, and “zone files” in BIND 8.x. These files are made known to the respective DNS server through a configuration file, named.conf. In earlier versions of BIND, it was called named.boot.

When fully formed, a host name is made up of a sequence of labels separated by dots. When read from right to left, as DNS parses it, it describes a path leading from the most inclusive domain in its tree, through successively more local domains, until its own host name is reached.

Using the full host domain name, this is how a DNS server traverses the DNS data base, starting at the right-most, most inclusive domain, following data maintained by the various DNS administrators in their respective data files, until it finds the target host name, and its IP address.

A domain name is also made up of a sequence of labels separated by dots. Rather than describing a host, it describes a domain, under which other sub-domains and/or hosts exist. It can be located in the DNS data base by DNS servers the same way as was the host domain name.

Sometimes a particular DNS server will not manage an entire domain. Rather, the domain will be broken up into pieces, called “zones”. Responsibility for these various zones is “delegated” to other DNS servers, and their respective DNS administrators. So, in DNS

configuration files, instead of describing a domain for which it is responsible, the more general term “zone” is used.

It is also common, in fact recommended, for a DNS Server to have at least one “backup”, another machine that will respond to queries when the main server is down. The main server is known as the “master” and the backup as the “slave”. In previous versions of BIND, they were known as “primary” and “secondary”.

The rest of this section concerns itself with only “leaf” DNS servers, that is, servers that only serve hosts. These servers have no domains under it, only hosts.

There are four types of db or zone files used by a DNS server, each identified in the server’s `named.conf` file:

- `zone.DOMAIN` — provides name-to-address mapping
- `zone.ADDR` — provides address-to-name mapping
- `zone.LOCAL` — a `zone.ADDR` file that provides loopback mapping
- `zone.CACHE` — a `zone.DOMAIN` file that identifies root name servers; also known as the “`zone.hint`” file.

Overview of DNS BIND/iX

In this implementation of BIND 8.1.1, the configuration and data files for the DNS server are found under the `/BIND/PUB` directory of the POSIX name space, though the DNS server is started by running a job from the MPE/iX name space — `JNAMED.PUB.BIND` which runs program `NAMED.PUB.BIND`.

The `NAMED` program maintains a cache of information, taken initially from its zone files (database files) augmented by information which it has retrieved from other DNS servers on the network.

Syslog is the standard event logging subsystem for UNIX, which has now been implemented as Syslog/iX on MPE/iX 6.0 running in the POSIX environment. DNS BIND/iX server logging is handled by Syslog/iX. In order to run the DNS server, you will first configure and start Syslog/iX. Details on configuring and running Syslog/iX can be found in Appendix E , “Configure and Run Syslog/iX,” of this manual.

The client (resolver) code has already been implemented in earlier releases of MPE/iX via library routines and the configuration file `RESLVCNF.NET.SYS` (linked to `/etc/resolv.conf`). `RESLVCNF.NET.SYS` contains information about the domain of the client host and the IP addresses of the local DNS servers who can be queried for information.

DNS BIND/iX Component Files

The major files for the implementation of DNS BIND/iX are found in `PUB.BIND` and `NET.SYS` in the MPE/iX name space, and under directories `/BIND/PUB` and `/etc` in the POSIX name space.

`JNAMED.PUB.BIN` The job which runs the DNS server.

`NAMED.PUB.BIND` The DNS server program.

`RESLVCNF.NET.SYS` The DNS client (resolver) configuration file. Linked to `/etc/resolv.conf`.

`/etc/resolv.conf` The DNS client (resolver) configuration file. Linked to `RESLVCNF.NET.SYS`.

`/BIND/PUB/etc/named.conf` The configuration file for the DNS server program. It used to be called `/etc/named.boot` in earlier versions of BIND.

`/BIND/PUB/etc/zone.<various>` The zone files contain the data which will be loaded into the DNS server's cache when it is started — these used to be called `db` or "database" files in earlier versions of BIND. They replace the `db.<various>` files.

Several example zone files have been included with the DNS BIND/iX product.

`/BIND/PUB/etc/nslookup.help` The help text for the `nslookup` utility.

`/BIND/PUB/bin/nslookup` Interactive name server query utility.

`/BIND/PUB/bin/dnsquery` DNS server query tool.

`/BIND/PUB/bin/host` Host information lookup tool.

`/BIND/PUB/bin/addr` Address lookup tool.

`/BIND/PUB/bin/named-bootconf.pl` Perl script to assist in converting BIND 4.x `named.boot` to 8.x `named.conf`.

`/BIND/PUB/bin/nsupdate` Zone transfer program — called internally by nameservers to transfer zone information from primary to secondary servers

`/BIND/PUB/public_html` Linked to sub-directory `/BIND/PUB/doc-8.1.1/html`

In addition, there are the following directories included with this product:

`/BIND/PUB/include` Include code files.

`/BIND/PUB/lib` Library routines

Server Configuration File `named.conf`

The configuration file, `named.conf`, has a completely new syntax. The configuration file in

BIND 4.x was called named.boot.

The utility “named-bootconf.pl”, written in Perl, available with the package, can be used to convert 4.x (8.1.1) configuration files. The complete path of this file in the installation is /BIND/PUB/bin/named-bootconf.pl.

See Appendix D, “Server Configuration Migration,” for directions on running the named-bootconf.pl utility.

The file named.conf provides configuration information about the database, information for the DNS server program NAMED. The database information is divided into zones. A zone will be either a domain (for example, india.hp.com) or an IP network (for example, 4.10.15.IN-ADDR.ARPA. A DNS server needs both types of zones in order to be able to resolve names to IP addresses, and IP addresses to names.

The named.conf configuration indicates to NAMED which zones it is going to be a server for, whether or not the server is a master or a slave for each zone, and points to the files where the database information is maintained. When a slave zone is configured, the address of the master server for that zone will also be included.

A DNS server which is the master for a zone is the one where the master copy of the data is maintained. A DNS server which is a slave for a zone may keep a copy of the data too, but will open a connection to the master server in order to obtain updates. This update process is called a “zone transfer”. A DNS server may be both the master server for some zones, and a slave server for others.

A template /BIND/PUB/etc/named.conf has been provided with the installation of DNS BIND/iX. You can use this file, following the commented instructions within it as a basis for your own /BIND/PUB/etc/named.conf. Advanced users may need to refer to Appendix B, “BIND 8 Configuration File,” for a complete list of directives that can be configured for BIND 8. The following is the template /BIND/PUB/etc/named.conf file:

```
options {
    directory "/BIND/PUB/etc";
    // The following is the IP address of the MPE/iX system that is running
    NAMED.
    // YOU MUST CHANGE THIS TO BE YOUR OWN IP ADDRESS!
    listen-on { nnn.nnn.nnn.nnn; };
};
/** List any servers here that you communicate with that are also running
BIND 8.1 or greater. Replace ALL OF THESE with your own servers, if any. */
server nnn.nnn.nnn.nnn {
    transfer-format many-answers;
};
// Defines the root. From ftp://rs/internic.net/domain/named.root.
zone "." {
    type master;
    file "zone.hint"
};
//      DNS optimiation tricks for "special" addresses. You will need to
//      edit all of these files to specify the hostname of your own
nameserver
//      and the e-mail address of the DNS maintainer.
zone "0.0.127.in-addr.arpa" {
```

```

        type master;
        file "zone.127.0.0";
};

zone "0.in-addr.arpa" {
    type master;
    file "zone.bogus.0";
};

zone "255.in-addr.arpa" {
    type master;
    file "zone .bogus.255";
};

// A master zone. Substitute one of your own zones here.
// Slave zones. Replace ALL OF THESE with your own.
zone "csy.hp.com" {
    type slave;
    file "zone.slave";
    master { nnn.nnn.nnn.nnn; nnn.nnn.nnn.nnn; }
};

```

Configuring Master Zones

A sample configuration unit for a master zone is shown here:

Example:

```

zone "43.10.15.IN-ADDR.ARPA" {
    type master;
    file "zone.15.10.43";
};

```

The file zone.15.10.43 will have entries like:

```

        IN      SOA      bindserver.india.hp.com.  bind_admin.india.hp.com. (
                                104      ; Serial
                                10800    ; Refresh every 3 hours
                                3600    ; Retry every hour
                                604800  ; Expire after a week
                                86400   ) ; Minimum ttl of 1 day
        IN      NS      bindserver.india.hp.com.

1      IN      PTR      m1.india.hp.com.
2      IN      PTR      m2.india.hp.com.
3      IN      PTR      m3.india.hp.com.
4      IN      PTR      m4.india.hp.com.
5      IN      PTR      m5.india.hp.com.

```

Configuring Slave Zones

A sample configuration unit for a slave zone is shown here:

```

zone "41.10.15.IN-ADDR.ARPA" {
    type slave;
    file "zone.15.10.41";
    masters {
        15.70.188.45;
    };
};

```

The IP address of the server that is primary for that domain is specified in the masters { } section of the configuration. There could be more than one master for a given zone.

When the nameserver comes up, looking at this configuration, it makes a connection with the nameserver running on 15.70.188.45 and does zone transfer, if required. It also makes a local copy of this file.

Data Files

The files that the primary nameservers load their zone data from are called data files or zone files. They are also referred to as db files, short for database files.

The data files contain resource records that describe the zone. The resource records describe all the hosts in the zone.

Root Cache Data (Hint File)

Besides your local information, the nameserver also needs to know where the nameservers for the root domain are. This information must be retrieved from the Internet host ftp.rs.internic.net.

Explaining DNS Database Files

This is a typical DNS zone.domain file for the domain maxx.net. (Its name would be zone.maxx.net. It will translate from a host name to its IP address.)

```
;
; Addresses for the local domain
maxx.net.      IN      SOA    nova.maxx.net. tyager.nova.maxx.net. (
                                9602171      ; Serial
                                36000         ; Refresh every 10 hours
                                3600          ; Retry after 1 hour
                                360000        ; Expire after 100 hours
                                36000         ; Minimum TTL is 10 hours )

; Define name servers
;
maxx.net.      IN      NS     nova.maxx.net.
maxx.net.      IN      A      204.251.17.241

; Define localhost
;
localhost     IN      A      127.0.0.1

; Set up hosts
;
maxx          IN      A      204.251.17.241
              IN      MX     5     nova.maxx.net.
maxx.net.     IN      MX     5     nova.maxx.net.
;
; All mail for net delivered to nova
```

```

;
;*          IN      MX      10    nova.maxx.net.
www        IN      CNAME    nova.maxx.net.
ftp        IN      CNAME    nova.maxx.net.
news       IN      CNAME    nova.maxx.net.
mail       IN      CNAME    nova.maxx.net.
ns         IN      CNAME    nova.maxx.net.
loghost    IN      CNAME    nova.maxx.net.
lucy       IN      A         204.251.17.242
linux      IN      CNAME    lucy.maxx.net.
lucy       IN      MX       10    lucy.maxx.net.
messdos    IN      A         204.251.17.243
messdos    IN      MX       10    messdos.maxx.net.
pentium    IN      CNAME    messdos.maxx.net.
solaris    IN      A         204.251.17.244
solaris    IN      MX       10    solaris.maxx.net.
maxx4      IN      CNAME    solaris.maxx.net.
maxx5      IN      A         204.251.17.245
maxx5      IN      MX       10    maxx5.maxx.net.
maxx6      IN      A         204.251.17.246
maxx6      IN      MX       10    maxx6.maxx.net.

```

Most database file entries are known as DNS resource records. Generally, the resource records are shown in order: SOA, NS, followed by the other types, but this ordering isn't required. The data in each entry may be entered in upper, lower, or mixed case. All entries in the database file must start at the beginning of the line. Blank lines as well as any text following a semicolon is ignored.

SOA stands for Start of Authority. This acronym notifies named that operational parameters follow. The most important one is the Serial field. Every time you make a change to a database file, you must increment its serial number. Only by doing this will secondary servers know they need to reach into your system and pull out new name server data, a procedure known as a "zone transfer." Many DNS administrators use a date-time stamp for this field, like 9602171 for the first version on February 17, 1996.

First, focus on the SOA section:

```
maxx.net. IN SOA nova.maxx.net. tyager.maxx.maxx.net.
```

The "maxx.net." field tells named the domain defined by this file. The name server will automatically append it to any host name that appears in the file. The trailing dot is not a type; it keeps named from trying to tack on your domain name. Without it, the resolver would be confused by named's expansion of my domain name to "maxx.net.maxx.net."

The IN stands for the "Internet" class of data. Even though other classes exist, they aren't in common usage. The "nova.maxx.net" field is the host on which these database files reside. Finally, "tyager.nova.maxx.net" represents the e-mail address of the DNS administrator, where the first dot (between tyager and nova) would be replaced by the @ symbol to create a valid address. (The @ symbol can't be used here because it has a reserved meaning in DNS database files.)

The open parenthesis at the end of the line allows you to split the SOA record across physical lines for readability:

```

          9602171          ; Serial
          36000           ; Refresh every 10 hours

```

```
3600           ; Retry after 1 hour
360000        ; Expire after 100 hours
36000         ; Minimum TTL is 10 hours )
```

The “serial” field was discussed earlier.

The remaining four fields specify various time intervals (all values in seconds) used by the secondary name server:

Refresh	The time interval that must elapse between each poll of the primary by the secondary name server (here 36,000 seconds or 10 hours). If the “serial number” has been updated on the primary, the secondary assumes its data is stale and requests updated information as a “zone transfer.”
Retry	The time interval used between successive connection attempts by the secondary to reach the primary name server in case the first attempt failed (here 3,600 seconds or one hour). Generally, less than the “refresh” time.
Expire	The time interval after which the secondary expires its data if it can’t reach the primary name server (here 360,000 seconds or 100 hours). The secondary will refuse to service requests after this interval.
Minimum	The minimum time-to-live value, which specifies how long other servers should cache data from the name server (here 36,000 seconds or 10 hours).

There are several types of resource records, identified by the key word in field three of each record. You may present records in any order, but try to organize them for clarity. The NS (name server) record tells the hosts that query your server where the name servers for this domain can be found:

```
maxx.net.      IN      NS      nova.maxx.net.
```

You must include in this list at least one name server, that is the name of the server specified in the SOA record. You can list multiple name servers for your domain. In fact, your domain should have at least two name servers. Your Internet service provider will probably allow you to use their name server as a secondary for your domain, but it must have the trailing dots!

```
maxx.net      IN      A      204.251.17.241
```

The first A record, which resolves a fully-qualified host name to an IP address, is a special one. It defines an IP address for unqualified queries, that is, queries for the host `maxx.net`.

Other A records like this one:

```
lucy          IN      A      204.251.17.242
```

provide name-to-address mapping for a specific named host. The domain defined in this file (`maxx.net`) is appended to the host name you show in the first field.

The CNAME records create aliases for existing hosts. These examples illustrate a few common uses:

```
www          IN      CNAME   maxx.maxx.net.
ftp          IN      CNAME   maxx.maxx.net.
```

You can give a host any alias you like, and as many aliases as you want. The host needn’t answer to that name, that is, the alias doesn’t need to be the host’s true name as reported by `hostname` or `uname`.

The other vital type of record is `MX`. This tells SMTP e-mail software where to send mail for each named host:

```
lucy      IN      MX      10   lucy.maxxx.net.
```

When a remote host's mail delivery program sees an e-mail address in your domain, it will query your name server for its applicable `MX` record or records. Every user on your LAN can receive e-mail, even if not every host is running its own e-mail software. The `MX` record for `lucy`, for instance, could easily redirect e-mail to another host on the LAN.

The number (10 in this case) in the fourth field represents a preference value. If you define multiple `MX` records for a host, delivery is attempted to lower-preference value hosts first. The actual value isn't important, only its relationship to other preference values.

On larger LANs it's a good idea to create backup e-mail servers. Smaller LANs can simply rely on the fact that most SMTP mailers will retry deliveries to the site for three days before returning a message to its sender.

The line, shown commented out here, would arrange to redirect e-mail for all hosts in this domain to a single machine:

```
;  
; All mail for net delivered to nova  
;  
;*      IN      MX      10   nova.maxxx.net.
```

This is a very good idea for LANs that benefit from a central e-mail repository.

Address-to-Name Mapping

Also called reverse mapping, the `zone.ADDR` db file allows resolvers to post queries armed with only the IP address of a host. This reverse mapping is used, for example, by Internet server software that prefers to log host names rather than less informative IP addresses.

Address-to-name mapping data will be provided for a DNS server by `PTR` entries in its `zone.ADDR` files, one for every network served by this DNS server, and its `zone.LOCAL` file.

Each entry will indicate the IP address in reverse order, then the host name. For example, for host `littledog.maxxx.net`, whose IP address is `204.251.17.249`, in the `zone.ADDR` file it's `PTR` entry would look like:

```
249.17.251.204.      IN      PTR      littledog.maxxx.net.
```

Why is it backwards? Recall that DNS does its parsing from right to left, from most inclusive to most specific. For IP addresses, it needs to parse in the same direction. But IP addresses, from right to left, go from most specific to most inclusive. So the simple answer is to reverse the IP address in the `NDS PTR` records. Now DNS can parse in the same direction, and resolve in the same order — from most inclusive to most specific.

A shortcut in `PTR` records is often used. It looks like this:

```
249      IN      PTR      littledog.maxxx.net.
```

If the dot is left off the IP address in the `PTR` record, DNS will complete the IP address with the IP address of the domain, specified in the file's `SOA` record. This is also true for `A` records in name-to-address mapping db files. If the dot is left off, DNS will automatically try to complete the name with the full domain name in this zone. Paying attention to the terminating dot is important.

For the `zone.LOCAL` file we describe the loopback address just as you would expect it, now that we know we have to reverse it. The PTR entry in the `zone.LOCAL` file would look like:

```
1.0.0.127.      IN      PTR      localhost.
```

or, using the shortcut:

```
1              IN      PTR      localhost.
```

Only one line from `named.conf` remains to be discussed, the “cache” entry. This is a bit of a misnomer as it doesn’t have anything to do with local caching. Instead, it defines the master root domain name servers for the Internet. You can retrieve this list from `ftp://nic.ddn.mil/netinfo/root-servers.txt`. You will need to check this site periodically to ensure you have the latest list.

This file lists the root domain servers in human-readable format. You’ll need to reformat it for consumption by `named`. Here’s what the cache file looks like:

```
;          Servers from the root domain
;          ftp://nic.ddn.mil/netinfo/root-servers.txt
;
.           999999999      IN      NS      A.ROOT-SERVERS.NET
.           999999999      IN      NS      B.ROOT-SERVERS.NET
.           999999999      IN      NS      C.ROOT-SERVERS.NET
.           999999999      IN      NS      D.ROOT-SERVERS.NET
.           999999999      IN      NS      E.ROOT-SERVERS.NET
.           999999999      IN      NS      F.ROOT-SERVERS.NET
.           999999999      IN      NS      G.ROOT-SERVERS.NET
.           999999999      IN      NS      H.ROOT-SERVERS.NET
.           999999999      IN      NS      I.ROOT-SERVERS.NET
;
; Root servers by address
A.ROOT-SERVERS.NET  999999999      IN      A 198.41.0.4
B.ROOT-SERVERS.NET  999999999      IN      A 128.9.0.107
C.ROOT-SERVERS.NET  999999999      IN      A 192.33.4.12
D.ROOT-SERVERS.NET  999999999      IN      A 128.8.10.90
E.ROOT-SERVERS.NET  999999999      IN      A 192.203.230.10
F.ROOT-SERVERS.NET  999999999      IN      A 192.5.5.241
G.ROOT-SERVERS.NET  999999999      IN      A 192.112.36.4
H.ROOT-SERVERS.NET  999999999      IN      A 128.63.2.53
I.ROOT-SERVERS.NET  999999999      IN      A 192.36.148.17
```

Here, the dot (.) refers to the root domain and the 999999999 means a very long time-to-live value. The TTL value is no longer used for caching because the data isn’t discarded if it times out, but administrators generally keep it around because it does no harm.

Your site may not have access to the Internet or may have protected its connection via a firewall. Often in this type of DNS configuration, one or more machines will be designated as a root server. In this case, the cache file will contain a list of internal root servers, and not the official Internet master root domain servers.

Testing Your Name Server

Perform simple checks on your name server’s health with `nslookup`. This utility is standard with every TCP/IP-network-aware version of UNIX. There are other similar tools available — see “List of Utilities” later in this section for details.

You can find the source code for dig at several anonymous FTP archive sites, including: ftp://ftp.wonderland.org/NetBSD/NetBSD-current/src/usr.sbin/named/dig/ for the NetBSD release. Use Archie to find other sites.

The nslookup utility can be used interactively, much like other programs, such as ftp. That is, if you invoke this program without command line arguments, it displays a prompt and waits for your command:

```
>server mpe3000
```

```
Default Name Server: mpe3000.cup.hp.com Address: 15.13.199.80
```

By default, nslookup performs queries based on host names you submit; just enter a host name after the prompt:

```
> romeo
Server:    mpe3000.cup.hp.com
Address:   15.13.199.80
```

```
Name:      romeo.cup.hp.com
Address:   15.13.194.242
```

```
> 15.12.194.242
Server:    mpe3000.cup.hp.com
Address:   15.13.199.80
```

```
Name:      romeo.cup.hp.com
Address:   15.12.194.242
```

You can check the resource records information about name server:

```
> set type=ns
> mpeworld
Name Server:  mpeworld.cup.hp.com
Address:     15.13.199.80

origin = dns.cup.hp.com
mail addr = dns-admin.dns.cup.hp.com
serial = 96092255
refresh = 10800 (3 hours)
retry = 3600 (1 hour) expire = 604800 (7 days)
minimum ttl = 86400 (1 day)
```

How to Run The DNS Server

1. Configure and start Syslog/iX see Appendix E , “Configure and Run Syslog/iX.”
2. Examine /BIND/PUB/etc/named.conf and customize for your own environment.
3. Configure the zone data files referenced in your /BIND/PUB/etc/named.conf.
4. Add your server’s IP address as the first nameserver entry in /etc/resolv.conf for all MPE and HPUNIX hosts that you wish to use this server for resolution queries. On MPE hosts, make sure that /etc/resolv.cnf is actually a symlink pointing to the real data at RESLVCNF.NET.SYS.
5. :stream JNAMED.PUB.BIND

6. Stop BIND by issuing the command `:ABORTJOB`.

Configuring the DNS Resolver

The file `RESLVCNF.NET.SYS` is the configuration file for the Domain Name resolver. It should be linked to `/etc/resolv.conf`. If the file does not already exist, then it can be copied from `RSLVSAMP.NET.SYS` to `RESLVCNF.NET.SYS` and then modified to contain information about your local domain and servers.

Each entry in the resolver file consists of a keyword followed by a value separated by white space. The keyword and its associated value must appear on a single line, and the keyword must start the line. Comment lines start with a pound sign (`#`) or semicolon (`;`).

`domain` Enter the default domain name. This string will be appended to queries passed to the local DNS server. The default names should be written without a trailing dot:

```
domain india.hp.com
```

It is important to get the syntax correct as the resolver does not report errors. If more than one instance of the domain keyword is present, the last instance will override. To specify multiple domains for an unqualified name lookup, use the search directive.

`search` The search directive is optional but overrides the domain directive for specifying which domains should be searched for unqualified host name lookups. You should add a search entry if users on a system commonly try to connect to nodes in another domain. The format is the search directive followed by up to six domains, separated by a white space.

```
search cup.hp.com hp.com
```

`nameserver` The nameserver directive tells the resolver the IP address of a name server to query. For example, the line:

```
nameserver 15.32.17.2
```

instructs the resolver to send queries to the name server running at IP address `15.32.17.2` instead of the local host.

The resolver will also allow you to specify up to three name servers using multiple nameserver directives. They will be tried in the order in which they appear in the `RESLVCNF` file, only passing to the next listed nameserver if the previous one is not responding. Note that the resolver will only query subsequent name servers if there is no response, if the previous nameserver has already replied that it cannot resolve a query, no further lookup will be attempted.

NOTE It is very important that you omit the leading zeros in the domain name resolver files. If you enter leading zeros here, the resolver routines will

interpret the numbers as octal numbers.

sortlist	<p>This directive is a mechanism which lets you specify subnets and networks for the resolver to prefer if it receives multiple addresses as a result of a query. The format is the sortlist directive, followed by a list of network addresses may also include a subnet mask, which immediately follows the address, preceded by a slash symbol (/).</p> <pre>sortlist 128.32.42.0/255.255.255.0 15.0.0.0</pre>
options	<p>The options directive lets you set two internal resolver settings.</p> <pre>options debug</pre> <p>The above directive will set an internal flag which causes debugging information to be produced on standard output.</p> <pre>options ndots:2</pre> <p>The above directive sets the minimum number of dots a domain name query must contain before the resolver will assume that it is a fully qualified name and therefore does not need to append the default domain (or searchlist argument) before sending it to the server.</p> <p>The options directive can combine both settings on the same line.</p> <pre>options debug ndots:2</pre>
lines	<p>Beginning with a pound sign (#) or a semicolon (;) in the first column, they are interpreted as comments and ignored by the resolver.</p>

List of Utilities

- nslookup — query Internet name servers interactively

Example:

```
* nslookup quasar.india.hp.com
Name Server: hpmpea2.cup.hp.com
Address: 15.61.192.116

Non-authoritative answer:
Name: quasar.india.hp.com
Address: 15.10.45.114
```

- dig — Domain Information Groper

Example: shell/iX> dig

```
; <<>> DiG 8.1 <<>>
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 12
;; QUERY SECTION:
```

```
;; ., type = NS, class = IN
;; ANSWER SECTION:
.                2d23h2m52s IN NS   japan.cns.hp.com.
.                2d23h2m52s IN NS   paloalto.cns.hp.com.
.                2d23h2m52s IN NS   singapore.cns.hp.com.
.                2d23h2m52s IN NS   andover.cns.hp.com.
.                2d23h2m52s IN NS   atlanta.cns.hp.com.
.                2d23h2m52s IN NS   bbnhs.cns.hp.com.
.                2d23h2m52s IN NS   boise.cns.hp.com.
.                2d23h2m52s IN NS   brahs.cns.hp.com.
.                2d23h2m52s IN NS   colorado.cns.hp.com.
.                2d23h2m52s IN NS   corvallis.cns.hp.com.
.                2d23h2m52s IN NS   cupertino.cns.hp.com.
.                2d23h2m52s IN NS   fortcollins.cns.hp.com.
.                2d23h2m52s IN NS   gvahs.cns.hp.com.
;; ADDITIONAL SECTION:
japan.cns.hp.com.      2d23h36s IN A    15.74.137.1
paloalto.cns.hp.com.  2d23h36s IN A    15.1.200.2
singapore.cns.hp.com. 2d23h35s IN A    15.43.40.31
andover.cns.hp.com.   2d23h35s IN A    15.4.152.7
atlanta.cns.hp.com.   2d23h36s IN A    15.24.240.5
bbnhs.cns.hp.com.     2d23h36s IN A    15.195.32.10
boise.cns.hp.com.     2d23h35s IN A    15.10.216.25
brahs.cns.hp.com.     2d23h36s IN A    15.195.104.10
colorado.cns.hp.com.  2d23h36s IN A    15.13.48.11
corvallis.cns.hp.com. 2d23h36s IN A    15.7.240.32
cupertino.cns.hp.com. 2d23h35s IN A    15.36.88.4
fortcollins.cns.hp.com. 2d23h35s IN A    15.6.184.40

;; Total query time: 0 msec
;; FROM: mpeworld to SERVER: default -- 0.0.0.0
;; WHEN: Mon May 18 22:15:45 1998
;; MSG SIZE sent: 17 rcvd: 494
```

- **host** — look up host names using domain server.

Example:

```
shell/iX> host quasar.india.hp.com
quasar.india.hp.com has address 15.10.45.114
quasar.india.hp.com mail is handled (pri=90) by hpmdd58.india.hp.com
quasar.india.hp.com mail is handled (pri=100) by palsmtp.hp.com
quasar.india.hp.com mail is handled (pri=150) by atlsmtp.hp.com
quasar.india.hp.com mail is handled (pri=10) by quasar.india.hp.com
quasar.india.hp.com mail is handled (pri=50) by fakir.india.hp.com
```

- **addr** — get address of host
- **dnsquery** — Give all the DNS details and Mail exchange records.

Example:

```
shell/iX> dnsquery quasar.india.hp.com
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45601
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 7, ADDITIONAL: 10
;; quasar.india.hp.com, type = ANY, class = IN
quasar.india.hp.com.      1D IN MX    50 fakir.india.hp.com.
quasar.india.hp.com.      1D IN MX    90 hpmdd58.india.hp.com.
quasar.india.hp.com.      1D IN MX   100 palsmtp.hp.com.
quasar.india.hp.com.      1D IN MX   150 atlsmtp.hp.com.
```

quasar.india.hp.com.	1D IN MX	10 quasar.india.hp.com.
quasar.india.hp.com.	1D IN A	15.10.45.114
india.hp.com.	1D IN NS	fakir.india.hp.com.
india.hp.com.	1D IN NS	cauvery.india.hp.com.
india.hp.com.	1D IN NS	valmki.india.hp.com.
india.hp.com.	1D IN NS	hpmdd58.india.hp.com.
india.hp.com.	1D IN NS	palrel2.hp.com.
india.hp.com.	1D IN NS	atlrel2.hp.com.
fakir.india.hp.com.	1D IN A	15.10.40.3
hpmdd58.india.hp.com.	1D IN A	15.70.168.58
palsmtp.hp.com.	8H IN A	156.153.255.242
palsmtp.hp.com.	8H IN A	156.153.255.226
atlsmtp.hp.com.	8H IN A	156.153.255.210
atlsmtp.hp.com.	8H IN A	156.153.255.202
quasar.india.hp.com.	1D IN A	15.10.45.114
cauvery.india.hp.com.	1D IN A	15.10.40.5
valmiki.india.hp.com.	1D IN A	15.17.112.100
sahana.india.hp.com.	1D IN A	15.10.43.22

NOTE In order to run the various utilities, you will need to modify your PATH variable, adding the following two directories:

/BIND/PUB/sbin

/BIND/PUB/bin

DNS and Electronic Mail

One of the advantages of the Domain Name System over host tables is its support of advanced mail routing. DNS offers a mechanism for specifying backup hosts for mail delivery. The mechanism also allows hosts to assume mail handling responsibilities for other hosts. This lets diskless workstations that don't run mailers, for example, have mail addressed to them processed by their server. These features give administrators more flexibility in configuring electronic mail on their network.

MX Records

DNS uses a single type of resource record to implement enhanced mail routing, the MX record. MX records specify a mail exchanger for a domain name, a host that will either process or forward mail for the domain name.

In order to prevent mail routing loops, the MX record has an extra parameter, besides the domain name of the mail exchanger, a preference value like: `peets.mpk.ca.us IN MX 10 relay.hp.com` specifies that `relay.hp.com` is a mail exchanger for `peets.mpk.ca.us` at preference value 10.

DNS BIND Troubleshooting Steps

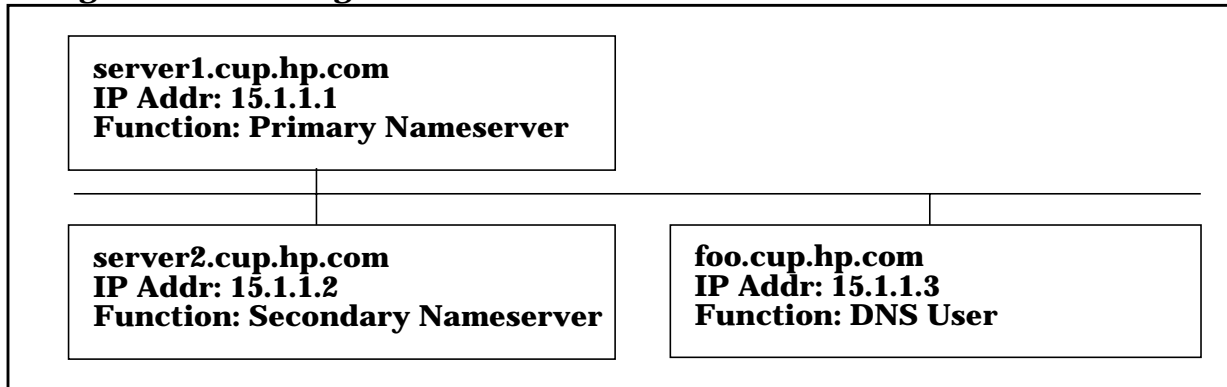
1. **Resources:** Find a resource who is experienced with DNS BIND/iX! If you're entering into this without DNS BIND/iX experience, you're off to a difficult start. Problems with this product are generally caused by poor configuration, so it's critical to have a DNS BIND literate engineering resource available for problem classification and management.
2. **Check the Obvious:** Those with experience in DNS BIND troubleshooting will have built up a number of quick "sanity checks" that they use. Often, these will result in a quick resolution without having to progress onto the next stages. If you don't have the experience (and can't find someone that does... recommended) or find that you're still unable to find the answer, you'll need to progress to the next steps.
3. **Detailed Problem Description:** Historical information is very valuable... is this a new DNS BIND installation, or has the site suddenly started to experience problems? No matter what the history, you will need to find out and document the exact symptoms being experienced.

It Used to Work: Find out if the DNS Administrator is aware of any configuration or network topology changes that could be tied to the recent DNS BIND problems. Make a note of anything they can suggest. Generally, these problems are caused by an incorrect configuration change, or some change in network topology, resulting in lost connectivity to systems required by the DNS environment (no route to a required system, an internal or external nameserver is down, system name/IP address change, poor configuration, and so forth).

New Configuration: In 99% of DNS BIND problems, the cause is poor configuration. Unfortunately, DNS is not an easy service for the novice to configure. There are many pitfalls waiting to trip a user. In a new configuration situation, you'll find the following steps will probably be needed.

4. **Topology Information:** Obtain and document a detailed description of the DNS topology used in this environment, Information on all the involved systems will be needed. It's important to be able to picture how all the systems connect to one another and the inter-dependencies any have with one another. If possible, an ASCII diagram of the topology is very often worth the effort (labeling each node with its system and DNS information, see Figure 8-1.).

Figure 8-1. Labeling Nodes



5. **Configuration Gathering:** Once you have a good understanding of the history, symptoms, and topology, it's time start examining the DNS configuration at the site. Relying on assumptions does not work with DNS BIND troubleshooting.

This information is needed from each system.

- From ALL Systems:

- a. Review the following files:

```

/etc/resolve.conf
/etc/nsswitch.conf (If present)
results for all lan interfaces
  
```

- b. Run the following commands:

```

nettool.net.sys "conf;summ;gui"
linkcontrol@,S
netcontrol <Niname>;STATUS for each appropriate NI
  
```

- c. From Nameservers:

```

All the information detailed above in "From ALL Systems"
/etc/named.conf (Or the customers equivalent)
The system's db files
  
```

Look in the `/etc/named.conf` file and the directory directive will tell you where to look for these. They are prefixed with `db` or `zone`, so may look like these examples:

`db.cache`, `db.root`, `db.127.0.0`, `db.cup`, etc.

6. **Configuration Validation:** Once the configuration information is gathered, it's time to sit down and wade through it all, looking for problems. By now you should have a good idea of how this DNS BIND topology fits together. Consider the symptoms, the history, the topology, and verify the levels of configuration that might be responsible for these problems.

Experience is the best tool, but there is one very good resource available that will help in troubleshooting DNS BIND:

DNS & BIND is a book written by Paul Albitz and Cricket Lui. The 2nd edition has recently been published, with some useful additions for the newer, post 4.8.3, versions of BIND (4.9.3 is covered in some detail). Published by O'Reilly & Associated, Inc. [2nd Edition ISBN: 1-56592-236-0]

7. **Troubleshooting Tools:** The following tools can be useful in troubleshooting DNS BIND problems:

nslookup (Available on all systems)

ping (Available on all systems)

Further information on the use of these tools can be found in the book *DNS & BIND*, as well as in the system man pages.

9 Apache for MPE/iX

Apache for MPE/iX is server software which turns your HP 3000 into a full-featured web server. With the Apache Webserver, HP 3000 users can now do business over the Internet.

As a web server, your HP 3000 can provide users with direct access to documents and applications residing on your system. These applications can include internet and intranet dynamic database connectivity using a browser as a common, easily maintained interface.

Introduction

Users make requests to the web server via a client browser using the Hypertext Transfer Protocol (HTTP). The client browser can be any one of a variety of browsers, including those from Microsoft and Netscape. The sole purpose of a web server is to translate the client's request (URL) into either a filename, and then send that file back over the network, or into a program name, and then run that program and send its output back.

Once you start the executable, HTTPD, Apache runs silently in the background, waiting for a client's request to arrive on a port to which it is listening. Apache listens on the port specified in its configuration file.

When a request arrives, Apache hands the request to one of its child processes to service and returns to listen again on the port.

Feature Set

Apache supports a rich set of features. The feature set for Apache running on MPE/iX is determined by the set of modules that are compiled into the Apache program. Additional features will be selected for future releases.

The current version of Apache for MPE/iX is based on Apache 1.3.4 from the Apache Software Foundation. In addition to the http core that is the heart of the Apache code, 28 more Apache modules are included in the Apache program. Some of the major features include:

The latest HTTP protocol, HTTP/1.1

Capabilities new to HTTP/1.1 include content negotiation (the server returns the data type and human language most preferred by the browser), persistent connections (the server uses the same socket for more than one request from the same client), and HOSTNAME variable in the request (for implementing virtual hosts on the server).

Advanced Logging

Apache on MPE/iX supports multiple log files, customized log files, and logging on events such as error status.

Access Control

Basic access control to resources, such as a directory, is provided through usernames and passwords. Access can also be limited by IP address, domain, or even by HTTP method (i.e. POST or GET)

Common Gateway Interface applications (CGI)

CGI is a mechanism for accessing external applications from the browser. These CGI applications can be written in any script or programming language which runs on MPE/iX. CGI provides dynamic output to the

user. That is, the application result is returned to the client at the time of invocation.

Server Side Includes (SSI)

SSI also provides dynamic output to a client. SSI is a set of commands that are embedded in an HTML page and are parsed and executed when the page is accessed. SSI commands include flow control statements, variable declaration, and execution of programs.

Cookies

Cookies are pieces of information generated by the web server and sent back to the browser for storage. For each subsequent request from the same client, the cookie is returned to the server. Cookies are useful for tracking what clients are accessing on a server.

Server-side Imagemaps

Server-side imagemaps are zones defined in an image that, when clicked, will send the client to a different URL.

URL Alias and Redirection

One part of the server's file system is mapped to another part when URL aliasing or redirection are used. This is useful for moving documents outside of the document tree.

Directory Indexing

For URLs ending in a "/" (a directory request), Apache will return either an index file or a directory listing. The behavior is determined by what is configured in the httpd.conf file.

Fix Typos in URLs

Apache will correct single character misspellings in a URL and will return the file it thinks you want.

As-is Files

Apache has the ability to send documents to a client without HTTP headers. This is useful for document writers who want to write their own headers.

To view the Apache version, run the program file with the -v option:

```
shell> /APACHE/PUB/HTTPD -v
Server version: Apache/1.3.4 (MPE-1.3.4-v1.0)
Server built:   Aug 24 1999 16:11:19
```

For a complete list of Apache modules, run the program file with the -l option:

```
shell> /APACHE/PUB/HTTPD -l
Compiled-in modules:
  http_core.c
  mod_env.c
```

```
mod_log_config.c
mod_log_agent.c
mod_log_referer.c
mod_mime_magic.c
mod_mime.c
mod_negotiation.c
mod_status.c
mod_info.c
mod_include.c
mod_autoindex.c
mod_dir.c
mod_cgi.c
mod_asis.c
mod_imap.c
mod_actions.c
mod_speling.c
mod_userdir.c
mod_alias.c
mod_access.c
mod_auth.c
mod_auth_anon.c
mod_cern_meta.c
mod_expires.c
mod_headers.c
mod_usertrack.c
mod_unique_id.c
mod_setenvif.c
```

Major Components

The major components for Apache are the web server program (HTTPD), the job stream file which runs the HTTPD program (JHTTPD), a set of configuration files for enabling Apache features, the `htdocs` directory containing HTML pages, the `logs` directory, and the `cgi-bin` directory. Apache also comes with a full, on-line manual set.

- HTTPD is the Apache web server program. “HTTP” stands for the protocol used between the client browser and the Apache web server. “D” stands for daemon, a system program which automatically handles certain system operations. The HTTPD web server program intercepts an incoming request from the browser, interprets and handles the request, then delivers output in the form of an HTML page to the client.

The HTTPD program file resides in the `/APACHE/PUB` directory and is linked with Privilege Mode (PM) capability. Because HTTPD is a privilege mode program, it must run under an MPE account and group which both have PM capability. PM is necessary for Apache to use port numbers less than 1024 (privileged ports). By default, Apache uses port 80.

- JHTTPD is a job stream file which runs the Apache web server as a standalone process. This file also specifies the system timezone and the location of the global Apache

configuration file, `httpd.conf`. JHTTPD resides in the `/APACHE/PUB` directory.

- The configuration files tell the HTTPD server program where to find files, which files and directories can be accessed, who can access them, and the location of executable programs on the server. These files give you the flexibility of having the web server do only what you want it to do. These files reside in `/APACHE/PUB/conf`.
- The `htdocs` directory contains the public documents, images, and data to be served to clients. The `htdocs` directory and the directories below it are available to anyone accessing your web server.
- The `logs` directory contains log data of both web server accesses and of errors.
- The `cgi-bin` directory contains CGI scripts which are executed by Apache on behalf of its clients.
- The `bin` directory contains a set of scripts and programs contributed by the Apache Software Foundation. They are included as examples only and are not supported by HP.
- The Apache manual set, residing in `/APACHE/PUB/htdocs/manual`, describes the entire Apache feature set and the syntax definition for configuring these features. Access to the manual documents is specified with the URL, `http://yourserver.com/manual/index.html`, where “yourserver.com” is the name of your HP 3000.

Hardware Requirements

The following are estimates for hardware resources required for an Apache installation.

- 32 MB of memory (64 MB recommended for machines with high traffic)
- 25 MB Disk Space (Apache directories and files + log files)
- Additional disk space for your documents.

The 25 MB of disk space would be used approximately in the following way:

- 5 MB for Apache and its assorted static files
- 10 MB for the access log (about 100,000 requests)
- 10 MB for the error log (about 100,000 errors)

If you are using less than the recommended disk space for the log files, you may want to monitor their size. Each request or error message is about 100 bytes and both logs, but especially the access log, can grow quite large. Each request to the web server creates one entry in the access log.

Steps for Starting Apache

To start the Apache web server:

1. Prepare your system for network access.
2. Copy and edit the configuration file.
3. Edit the JHTTPD job stream file.
4. Start the HTTPD web server program.

5. Verify that the HTTPD web server is running.

Prepare Your HP 3000 for Network Access

Before your HP3000 can act as a web server, it must be available for network access via TCP/IP. In preparation, you should:

- Configure TCP/IP on your system
- Have a domain name associated with your system's IP address

Apache communicates on the network using the HTTP Hypertext Transfer Protocol which, in turn, uses TCP/IP. NS Transport (the TCP/IP transport subsystem) is configured on your HP 3000 using NMMGR. In NMMGR, configure your system's IP address and subnet mask in screen NEXTPORT.NI.Niname.PROTOCOL.IP. TCP should also be configured with the recommended values in the table below using screen NEXTPORT.GPROT.TCP. Information on the TCP/IP parameters is available in the *NS3000/iX NMMGR Screens Reference Manual* from <http://docs.hp.com/mpeix>.

TCP/IP Parameter	Recommended Value
Maximum number of connections	20,000
Retransmission Interval Lower Bound	1 second
Maximum time to wait for remote response	120 seconds
Initial Retransmission Interval	2 seconds
Maximum Retransmissions per Packet	6
Connection Assurance Interval	120 seconds
Maximum Connection Assurance Retransmissions	2

After completing your system's TCP/IP configuration, run `:NETCONTROL START` from the CI command line and verify that it ran successfully. Also verify that your system can respond over the network by running ping either from an HP 3000 or another system:

```
:run ping.net.sys;info="15.99.200.390"
```

```
64 byte(s) from $0F0DC0CF : icmp seq = 11, time = 2 ms  
64 byte(s) from $0F0DC0CF : icmp seq = 12, time = 3 ms  
64 byte(s) from $0F0DC0CF : icmp seq = 13, time = 2 ms  
< CONTROL-y >
```

```
C:\>ping yourserver.com
```

```
Pinging yourserver.com [15.99.200.390] with 32 bytes of data
```

```
Reply from 15.99.200.390: bytes=32 time<10ms TTL=199  
Reply from 15.99.200.390: bytes=32 time<10ms TTL=199  
Reply from 15.99.200.390: bytes=32 time<10ms TTL=199
```

You will also want to get a domain name. This is a unique identifier such as "yourserver.com" which is used (instead of the IP address) to direct requests from a browser to your server. Request a domain name from the administrator of the Domain

Name Server (DNS) on your network.

Configure Apache

Apache reads two global configuration files when it starts: `httpd.conf` and `mime.types`. These configuration files determine how Apache behaves. Earlier versions of Apache read two additional global configuration files: `access.conf` and `srn.conf`. These additional global configuration files can still be used, but by default they are empty and their original content is now included in the `httpd.conf` file.

Edit your Configuration File

The `mime.types` file comes as `mime.types.default` and must be copied or renamed to `mime.types`. Similarly, the `magic.default` file must be copied or renamed. The `httpd.conf` file comes in two different versions. The `httpd.conf.default` file is for a UNIX-based platform installation. The `httpd.conf.sample` file was derived from the `httpd.conf.default` file with modifications for MPE. This is the file you should edit.

```
shell> cd /APACHE/PUB/conf
shell> cp mime.types.default mime.types
shell> cp magic.default magic
shell> cp httpd.conf.sample httpd.conf
shell> vi httpd.conf
```

1. Uncomment and replace “`yourserver.com`” with your own server’s name for the `ServerAdmin` and `ServerName` directives. You may also choose to make other changes. The Apache Server Project web site, <http://www.apache.org/docs>, has information about all of the configuration directives.

All lines beginning with `#` are comments and are ignored by the HTTPD program file. Also note that the content of the configuration files is case sensitive.

Changes to the global configuration files do not take effect until the web server is started (or stopped and restarted if it is already running).

The `httpd.conf` file could, alternatively, be edited with a PC editor by using SAMBA and setting up a share for the `/APACHE/PUB/conf` directory.

2. Verify the configuration file. It is a good idea to verify your configuration files before trying to start the web server. This verification is for syntax checking only.

```
shell>/APACHE/PUB/HTTPD -t
```

Edit the JHTTPD Job Stream File

The JHTTPD Job Stream File is used to run the HTTPD web server program in standalone mode with your local timezone:

```
!job jhttpd,www.apache,pub;outclass=,2
!setvar TZ `PST8PDT`
!run HTTPD;info='-f /APACHE/PUB/conf/httpd.conf'
!eoj
```

The timezone variable, `TZ`, should be set to your local timezone (for example, `EST5EDT` for Eastern Daylight Time, `PST8PDT` for Pacific Daylight Time, and `MST7MDT` for Mountain

Daylight Time). For more information about setting the timezone value, enter “man timezone” in the POSIX shell to read the POSIX help file.

To preserve the access permissions (ACD) on the JHTTPD file, you can edit the file using the secure feature of EDITOR:

```
:EDITOR
/t JHTTPD
/m ...
/set secure
/k
JHTTPD,UNN
JHTTPD ALREADY EXISTS - RESPOND YES TO PURGE OLD AND KEEP NEW
PURGE OLD?yes
/e
```

The JHTTPD file does not come with a password. If desired, the JOBSECURITY command can be used to provide password protection without adding your password to the file.

Start the HTTPD Web Server Program

The HTTPD web server program is started using the JHTTPD job file. Apache can be started from either the CI or the POSIX shell:

```
:STREAM JHTTPD.PUB.APACHE

or

:xeq sh.hpbin.sys -L
shell>callci "stream jhttpd.pub.apache"
```

Verify that Apache is Running

There are a number of ways to verify if the Apache web server is running or, if it is not, to isolate how far the startup process progressed.

After streaming the JHTTPD file, use :SHOWJOB to view the running job:

```
JOBNUM  STATE  IPRI  JIN  JLIST      INTRODUCED  JOB NAME
#J16    EXEC           10S LP      TUE 10:27A  JHTTPD,WWW.APACHE
```

Another method is to check server status using either ps from the POSIX shell or using :SHOWPROC at the CI. The parent process is the HTTPD process with the -f option.

```
:SHOWPROC; job=jhttpd,www.apache
D202   0:00.079      WAIT   J16    73   :RUN HTTPD;info='-f
/APACHE/PUB/conf/httpd.conf '
D238   1:25.371      WAIT   J16    97   (HTTPD.PUB.APACHE) -f
/APACHE/PUB/conf/httpd.conf
D202   0:00.066      WAIT   J16    78   (HTTPD.PUB.APACHE)
D202   0:00.072      WAIT   J16    76   (HTTPD.PUB.APACHE)
D202   0:00.069      WAIT   J16    93   (HTTPD.PUB.APACHE)
D202   0:00.090      WAIT   J16   106   (HTTPD.PUB.APACHE)
D202   0:00.050      WAIT   J16    99   (HTTPD.PUB.APACHE)
D202   0:00.055      WAIT   J16   107   (HTTPD.PUB.APACHE)

shell> ps -ef | grep HTTPD
```



```

      UID          PID          PPID  C   STIME TTY          TIME COMMAND
WWW.APACHE      720972          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE
WWW.APACHE      589902          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE
WWW.APACHE     3014749          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE
WWW.APACHE      655457          5963849  0   Dec 31 ldev10    1:23 HTTPD.PUB.APACHE info=-f
WWW.APACHE     4587619          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE
WWW.APACHE      262250          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE
WWW.APACHE     3407979          655457  0   Dec 31 ldev10    0:00 HTTPD.PUB.APACHE

```

After the HTTPD program is running, make sure you can also access files in the directory tree. Some pages to try are:

- **Your server's Home Page.** This brings up the default home page. To modify this page, edit the `/APACHE/PUB/htdocs/index.html` file.
`http://yourserver.com`
- **MGR.APACHE's Home Page.** This is the default page for user MGR.APACHE. To modify this page, copy the sample file, then edit `/APACHE/PUB/public_html/index.html`.

```

shell> cd /APACHE/PUB/public_html
shell> cp index.html.sample index.html
http://yourserver.com/~MGR.APACHE

```
- **The Apache on-line documentation manual**
`http://yourserver.com/manual`

If you are unsuccessful in starting the HTTPD program, you can get more information about the problem by trying one or more of these troubleshooting techniques:

1. Look at the output of the `JHTTPD` spoolfile
2. Check the messages in the `/APACHE/PUB/logs/error_log` file
3. Verify the syntax of the `httpd.conf` file. This catches many, but not all, syntax problems in the `httpd.conf` file

```

:run HTTPD.PUB.APACHE:info="-t"

```
4. Try to telnet to Apache's port (port 80 if you have not modified the port directive set in the `httpd.conf.sample` file).

If you cannot telnet to Apache's port, then your problem is not with the web server. You are not yet reaching your HP 3000 box. A successful telnet connection should look something like,

```

telnet yourserver.com 80

Trying...
Connected to yourserver.com.
Escape character is '^]'.

```

Stopping Apache

Apache is stopped by aborting the Apache job from either the CI or the POSIX shell. Use `SHOWJOB` to view the Apache job:

```
JOBNUM  STATE IPRI JIN  JLIST      INTRODUCED  JOB NAME
#J16    EXEC          10S LP          TUE 10:27A  JHTTTPD,WWW.APACHE
:ABORTJOB #J16 or :ABORTJOB JHTTTPD,WWW.APACHE
```

Apache can also be stopped using `:ABORTJOB` from the POSIX shell

```
:xeq sh.hpbin.sys -L
shell>callci "abortjob #j16" or shell>callci "abortjob jhttp,www.apache"
```

Error Logging

Apache error logging is useful when trying to start Apache as well as for monitoring a running web server. Apache logs errors into a log file called `error_log` by default. This log file resides in the `/APACHE/PUB/logs` directory.

The number of messages logged in the error log is set by the `LogLevel` directive in the `httpd.conf` file. Possible values for `LogLevel` (by increasing significance) are `debug`, `info`, `notice`, `warn`, `error`, `crit`, `alert`, `emerg`. When a particular level is specified, messages from all other levels of higher significance are reported as well. For example, when `LogLevel` is "info", then messages with log levels of `notice`, `warn` and up to `emerg` are also posted.

By default, the level is set to `warn`. Using a level of at least `crit` is recommended.

Adding Documents

Add new documents by creating them under the document root, `/APACHE/PUB/htdocs`. Documents can also be added outside of the document root using the `Alias` directive in the `httpd.conf` file or by using symbolic links.

The web server children that handle user requests run as `WWW.APACHE`, the username specified in the `JHTTTPD` file. For more security, user `WWW.APACHE` has been created with minimal capabilities. Make sure files are readable by `WWW.APACHE` and `cgi` scripts and programs are executable by `WWW.APACHE`. `WWW.APACHE` must also have traverse directory (TD) permission for all the directories in the path of the file that it is accessing. In the POSIX shell, traverse directory permission is shown as execute (x) permission for that directory.

```
cd htdocs
mkdir newdir
chown MGR.APACHE:APACHE newdir
chmod 750 newdir
```

One way to check if a file is accessible by Apache is to logon as `WWW.APACHE` and try to access the file.

Troubleshooting

For any kind of trouble with Apache, first look in the error log. Execute a `tail` command on the `error_log` from the POSIX shell to look at the last few entries. The last entry will be the most recent entry:

```
shellix> tail /APACHE/PUB/logs/error_log
```

For troubleshooting Apache at the source code level, the Apache program file can be run

with the MPE debugger. It is best to run it with the `-X` (capital "X") option to prevent the parent Apache process from creating child processes:

```
:run httpd.pub.apache;info="-X";debug
```

Unsupported Functionality

HP does not support Apache binaries built by individuals or organizations outside of HP.

HP also does not support the scripts and programs in the `/APACHE/PUB/bin` directory. This is contributed software from the Apache Software Foundation and is provided for example only.

Performance

For best performance, files returned to the user should be in bytestream format. For example, `.html`, `.htm`, `.shtml`, `.shtm`, `.txt`, `.gif`, `.jpeg`, and `.jpg` files, should be in bytestream format instead of in MPE-type format. Bytestream files are more compatible with Apache and with other POSIX applications than are MPE-type files. If you have a web page that calls many images which are not in bytestream format (BA), you could have noticeable performance degradation.

If any of your files under the document root (`htdocs`) are either MPE fixed ascii (FA), MPE variable ascii (VA), or MPE variable binary (VB) files, you should consider converting them to bytestream files using the `"tobyte"` utility. Program files (fixed binary (FB) files with an NMPROG filecode) should never be converted.

A file's filetype can be determined using either the POSIX `file` command or the CI `listfile` command:

```
shell/iX> file index*.html
index.html:  MPE/iX 256-byte variable length binary (filecode: 0)
index1.html: MPE/iX 80-byte fixed length ascii (filecode:0)

shell/iX>callci listfile ./index.html,2
PATH= /APACHE/PUB/htdocs/
CODE  -----LOGICAL RECORD-----  ----SPACE----  FILENAME
      SIZE  TYP      EOF  LIMIT R/B  SECTORS #X MX  FILENAME
      128W  VB       19   204800  1     32    1  8   index.html
      80B   FA       54   204800  1     32    1  8   index1.html
```

To convert an ASCII-type file (`.htm*`, `.shtm*`, or `.txt`), use the `tobyte` utility with the `-at` option. If it is a binary-type file (such as `.jpeg`, `.jpg`, or `.gif`), do not use the `-at` option:

```
tobyte -at /APACHE/PUB/htdocs/index.html /APACHE/PUB/htdocs/newindex.html
```

For more information on the `"tobyte"` utility, use the POSIX help facility (i.e. `man tobyte`).

If your Apache web server seems slow in responding, you might try running the Apache job stream file, `JHTTTPD`, in the C queue instead of in the default D queue. The changes shown below allow Apache to run in the C queue while keeping the default execution level for jobs in the D queue. The `jobpri` command can be executed on the console or in a `systart` file.

```
!job JHTTTPD,www.apache;pri=cs;outclass=,2
jobpri cs
```

Additional Documentation

Much of the public information available on Apache can be used for administrating Apache on MPE/iX. This especially applies to the description and usage of the over 128 different Apache configuration directives.

Sources for additional information include

- The Apache on-line manual pages included as part of MPE FOS at <http://yourserver.com/manual/index.html>
- The Apache Software Foundation's on-line documentation at <http://www.apache.org/docs>
- Apache books, published by various publishers, such as O'Reilly and Associates, Inc. and IDG Books Worldwide, Inc.

For downloadable software to enhance your web site (perl, sendmail, python, etc.), visit MPE's external Jazz web server at <http://jazz.external.hp.com>

A Samba/iX Sample Configuration File

The following is the sample configuration file `samp-smb.cnf` for Samba/iX that you can find in the `/usr/local/samba/lib` directory on the HP 3000 system:

```
# Sample config file for Samba/iX 0.7 and later
# Copy this file to /usr/local/samba/lib/smb.conf and adjust as needed.
# You must at least adjust the "interfaces" directive to match
# your IP address and subnet mask (if used) as the current version
# of Samba/iX is unable to retrieve the NMMGR configured values.

# Some of the directives in this sample file are redundant because
# they explicitly specify hardcoded default values that would also
# be in effect if the directives were omitted. They are nevertheless
# included here to document their availability for customization.

# IMPORTANT WARNING: Some of the configuration options do have serious
# security implications and can cause risks or security holes if used
# improperly, especially when you decide to run the SMBD job under a
# user with PM (or even SM) capabilities or even select an SM capable
# user in the "guest account" directive.

# The documentation for smb.conf (available as man page in ../docs as
# well as HTML file in ../html) is thus STRONGLY RECOMMENDED reading!

# Also see the installation and configuration instructions for the
# different ways of running SMBD (i.e. with or without a PM user and
# even without PM program capabilities at all) and the associated
# tradeoffs between feature sets and security issues.

# -----

# GLOBAL section (general parms and defaults for other sections)
```

```
[global]

# you MUST supply IP address and subnet mask of your 3000 here

interfaces = 12.34.56.78/255.0.0.0

# config file and log file used by smbd and nmbd are typically
# specified as command line options, unless you are using macros
# like eg %S or %m to get different files for each service or
# client machine, which allows very sophisticated (albeit complex)
# configurations (also see "include" directive and smb.conf doc)

# config file = /usr/local/samba/lib/smb.conf
# log file = /usr/local/samba/var/log.smb

# mapping of incoming usernames is possible and may e.g. be used
# to allow clients using Unix or PC style names like root or lappel
# instead of MPE style names like manager.sys or lars.appel

# multiple alias names are possible e.g. lars.appel = lappel lars

username map = /usr/local/samba/lib/user.map

# printcap file lists printer names for use by [printers] section

printcap name = /usr/local/samba/lib/printcap

# how much detail you want in the logfile (try 3 or 5 or higher)

debug level = 1

# can use a shell script if system does not supply statfs() routine

# dfree command = /usr/local/samba/lib/myfree

# used in conjunction with printcap file and [printers] section

load printers = yes
```

```
# the workgroup that your server belongs to

workgroup = SambaIX

# these can be used e.g. to create logon/logoff like console messages

# preexec = callci /usr/local/samba/lib/telnet tcon %S %u %m %I
# postexec = callci /usr/local/samba/lib/telnet tdis %S %u %m %I

# shares may be configured to accept connections without a validated
# user id and password (similar to anonymous ftp) and then assume the
# guest logon identity for accessing files and printers

guest account = mgr.samba

# -----

# PRINTERS section (optional but useful)

# This section work in conjunction with the printcap file and allows
# to configure a large number of printer shares without having to add
# separate detailed sections for each of them. The printer names and
# optional aliases are listed in the printcap file and the config parms
# are defined here. Special printers can still be defined explicitly.

# Directive "load printers" makes all entries available for browsing.
# Directive "auto services" allows a more selective browse offering.

[printers]

# only want printer shares shown, not the [printers] section itself

browseable = no

# enable this service for printing but not for file access

print ok = yes
```

```
write ok = no

# current version has problems with printing for non-guest users

guest ok = yes
guest only = yes

# the "staging" directory for print requests

path = /usr/local/samba/spool

# permissions will be more meaningful when non-guest printing works

create mode = 0700

# the lp family of print command only work as of MPE/iX release 5.5
# the rawlp utility sends file contents to spooler like "lp -oraw"

print command = /usr/local/samba/lib/rawlp %s %p ; rm %s

# -----

# HOMES section (optional but sometimes useful)

# This section provides access to user's home directories without
# having to add a separate section for each of them. The share name
# is considered to be a valid user id and the path defaults to that
# user's home directory. The share is created "on the fly" by using
# attributes from this section.

# Notice that home directories on MPE/iX are currently MPE groups
# and grant CD and TD permissions to every user (not just the user
# who belongs to this home group). This is equivalent to LISTFILE
# ability across the whole system (at least on group levels). Read
# or write access are nevertheless controlled by file system plus
# smb.conf security definitions.

# Notice further that either the connecting user or the user derived
```



```
# from the share name may be validated by the appropriate passwords.
# Thus it is possible e.g. for user lars.appel to connect to the home
# directory of manager.sys - with access rights bound by file system.

# Confusing, isn't it? -- You might want to comment out [homes] thus.

[homes]

# only want home share shown, not the [homes] section itself

browseable = no

# allowing guest logon is usually not desired for home directories

guest ok = no

# write access is usually desired for home directories but keep in
# mind that there is also the file system permissions that decide
# if the connecting user (validated by password) may read or write

write ok = yes

# this one attempts to restrict "cross access" e.g. the user lars.appel
# to the home of manager.sys -- but may cause problems for some clients

valid users = %S

# -----

# OTHER sections (explicit definitions of file or printer shares)

# The writable shares are placed under an MPE group with space limit

[temp]

# multiple users share one server directory but independent file
# ownership is maintained so that they might be able to "see" other
# users' files but still be unable to get read or write access
```

```
comment = Shared temp space for non-guest users
```

```
guest ok = no
```

```
write ok = yes
```

```
path = /SAMBA/SHR/temp
```

```
[public]
```

```
# multiple users share one server directory but file ownership is  
# forced to the guest logon identity resulting in every user being  
# able to "see" as well as read or write the other users's files
```

```
comment = Shared space with all users forced to guest
```

```
guest ok = yes
```

```
guest only = yes
```

```
write ok = yes
```

```
path = /SAMBA/SHR/public
```

```
[sambadoc]
```

```
comment = Samba doc files (readonly but guest allowed)
```

```
guest ok = yes
```

```
write ok = no
```

```
path = /usr/local/samba/docs
```

```
[sambahtm]
```

```
comment = Samba HTML files (readonly but guest allowed)
```

```
guest ok = yes
```

```
write ok = no
```

```
path = /usr/local/samba/html
```

B BIND 8 Configuration File

The following is a dummy configuration file example. This explains in brief what each configuration directive is useful for and its syntax. All the directives are not required for a typical BIND configuration.

```
/*
 * This is a worthless, nonrunnable example of a named.conf file that has
 * every conceivable syntax element in use. We use it to test the parser.
 * It could also be used as a conceptual template for users of new features.
 */

/*
 * C-style comments are OK
 */

// So are C++-style comments
# So are shell-style comments

// watch out for ";" -- it's important!
options {
    directory "."; // use current directory
    named-xfer "/usr/libexec/named-xfer"; // _PATH_XFER
    dump-file "named_dump.db"; // _PATH_DUMPFILE
    pid-file "/var/run/named.pid"; // _PATH_PIDFILE
    statistics-file "named.stats"; // _PATH_STATS
    check-names master fail;
    check-names slave warn;
    check-names response ignore;
    datasize default;
    stacksize default;
    coresize default;
    files unlimited;
    recursion yes;
    fetch-glue yes;
    fake-iquery no;
    notify yes; // send NOTIFY messages. You can

set // notify on a zone-by-zone
    // basis in the "zone" statement
    // see (below)
    auth-nxdomain yes; // always set AA on NXDOMAIN.
    // don't set this to 'no' unless
    // you know what you're doing --

older // servers won't like it.
    multiple-cnames no; // if yes, then a name may have more
    // than one CNAME RR. This use
    // is non-standard and is not
    // recommended, but it is available
```

```

supported // because previous releases
sites // it and it was used by large
// for load balancing.
allow-query { any; };
allow-transfer { any; };
transfers-in 10; // DEFAULT_XFERS_RUNNING, cannot
be // set > than MAX_XFERS_RUNNING
(20)
transfers-per-ns 2; // DEFAULT_XFERS_PER_NS
transfers-out 0; // not implemented
max-transfer-time-in 120; // MAX_XFER_TIME; the default
number // of minutes an inbound zone
transfer // may run. May be set on a
per-zone // basis.
/*
* The "transfer-format" option specifies the way outbound zone
* transfers (i.e. from us to them) are formatted. Two values are
* allowed:
*
* one-answer Each RR gets its own DNS message.
* This format is not very efficient,
* but is widely understood. All
* versions of BIND prior to 8.1 generate
* this format for outbound zone
* and require it on inbound transfers.
*
* many-answers As many RRs as will fit are put into
* each DNS message. This format is
* the most efficient, but is only known
* to work with BIND 8. Patches to
* BIND 4.9.5 named-xfer that enable it
* to understand 'many-answers' will be
* available.
*
* If you are going to be doing zone transfers to older servers, you
* shouldn't use 'many-answers'. 'transfer-format' may also be set
* on a host-by-host basis using the 'server' statement (see below).
*/
transfer-format one-answer;
query-source address * port *;
/*
* The "forward" option is only meaningful if you've defined
* forwarders. "first" gives the normal BIND
* forwarding behavior, i.e. ask the forwarders first, and if that
* doesn't work then do the full lookup. You can also say
* "forward only;" which is what used to be specified with
* "slave" or "options forward-only". "only" will never attempt
* a full lookup; only the forwarders will be used.
*/
forward first;
forwarders { }; // default is no forwarders

```

```

/*
 * Here's a forwarders example that isn't trivial
 */
/*
forwarders {
    1.2.3.4;
    5.6.7.8;
};
*/
topology { localhost; localnets; }; // prefer local nameservers
/*
 * Here's a more complicated topology example; it's commented out
 * because only one topology block is allowed.
 * topology {
    10/8; // prefer network 10.0.0.0
        // netmask 255.0.0.0 most
    !1.2.3/24; // don't like 1.2.3.0 netmask
        // 255.255.255.0 at all
    { 1.2/16; 3/8; }; // like 1.2.0.0 netmask 255.255.0.0
        // and 3.0.0.0 netmask 255.0.0.0
        // equally well, but less than 10/8
};
*/
listen-on port 53 { any; }; // listen for queries on port 53 on
// any interface on the system
// (i.e. all interfaces). The
// "port 53" is optional; if you
// don't specify a port, port 53
// is assumed.

/*
 * Multiple listen-on statements are allowed. Here's a more
 * complicated example:
 */
/*
listen-on { 5.6.7.8; }; // listen on port 53 on interface
// 5.6.7.8
listen-on port 1234 { // listen on port 1234 on any
    !1.2.3.4; // interface on network 1.2.3
    1.2.3/24; // netmask 255.255.255.0, except for
}; // interface 1.2.3.4.
*/

/*
 * Interval Timers
 */
clean-interval 60; // clean the cache of expired RRs
// every 'clean-interval' minutes
interface-interval 60; // scan for new or deleted interfaces
// every 'interface-interval' minutes
statistics-interval 60; // log statistics every
// 'statistics-interval' minutes
};

zone "master.demo.zone" {
    type master; // what used to be called "primary"
    file "master.demo.zone";
check-names fail;
allow-update { none; };

```

```

allow-transfer { any; };
allow-query { any; };
// notify yes;                                // send NOTIFY messages for this
                                              // zone? The global option is used
                                              // if "notify" is not specified
                                              // here.

also-notify { };                             // don't notify any nameservers other
                                              // than those on the NS list for this
                                              // zone

};

zone "slave.demo.zone" {
    type slave;                               // what used to be called "secondary"
    file "slave.demo.zone";
    masters {
        1.2.3.4;                               // where to zone transfer from
        5.6.7.8;
    };

    check-names warn;
    allow-update { none; };
    allow-transfer { any; };
    allow-query { any; };
    max-transfer-time-in 120;                 // if not set, global option is used.
also-notify { };                             // don't notify any nameservers other
                                              // than those on the NS list for this
                                              // zone

};

zone "stub.demo.zone" {
    type stub;                                // stub zones are like slave zones,
                                              // except that only the NS records
                                              // are transferred.

    file "stub.demo.zone";
    masters {
        1.2.3.4;                               // where to zone transfer from
        5.6.7.8;
    };

    check-names warn;
    allow-update { none; };
    allow-transfer { any; };
    allow-query { any; };
    max-transfer-time-in 120;                 // if not set, global option is used.
};

zone "." {
    type hint;                                // used to be specified w/ "cache"
    file "cache.db";
};

acl can_query { !1.2.3/24; any; };           // network 1.2.3.0 mask 255.255.255.0
                                              // is disallowed; rest are OK
acl can_axfr { 1.2.3.4; can_query; };       // host 1.2.3.4 and any host allowed
                                              // by can_query are OK

zone "non-default-acl.demo.zone" {
    type master;
    file "foo";
    allow-query { can_query; };
    allow-transfer { can_axfr; };
    allow-update {

```

```

        1.2.3.4;
        5.6.7.8;servers.
    };
};

key sample_key {
    algorithm hmac-md5;           // for TSIG; supported by parser
    secret "your secret here";   // but not yet implemented in the
                                // rest of the server
};

key key2 {
    algorithm hmac-md5;
    secret "ereh terces rouy";
};

server 1.2.3.4 {
    bogus no;                    // if yes, we won't query or listen
                                // to this server
    transfer-format one-answer;  // set transfer format for this
                                // server (see the description of
                                // 'transfer-format' above)
                                // if not specified, the global option
                                // will be used
    transfers 0;                // not implemented
    keys { sample_key; key2; }; // for TSIG; supported by the parser
                                // but not yet implemented in the
                                // rest of the server
};

logging {
    /*
     * All log output goes to one or more "channels"; you can make as
     * many of them as you want.
     */

    channel syslog_errors {      // this channel will send errors or
        syslog user;            // or worse to syslog (user facility)
        severity error;
    };

    /*
     * Channels have a severity level. Messages at severity levels
     * greater than or equal to the channel's level will be logged on
     * the channel. In order of decreasing severity, the levels are:
     *
     *     critical          a fatal error
     *     error
     *     warning
     *     notice            a normal, but significant event
     *     info              an informational message
     *     debug 1          the least detailed debugging info
     *     ...
     *     debug 99         the most detailed debugging info
     */

    /*
     * Here are the built-in channels:
     */
    channel default_syslog {
        syslog daemon;
        severity info;
    };
};

```

```

*     };
*
*     channel default_debug {
*         file "named.run";
*         severity dynamic; // this means log debugging
*                           // at whatever debugging level
*                           // the server is at, and don't
*                           // log anything if not
*                           // debugging
*     };
*
*     channel null { // this is the bit bucket;
*         file "/dev/null" // any logging to this channel
*                           // is discarded.
*     };
*
*     channel default_stderr { // writes to stderr
*         file "<stderr>"; // this is illustrative only;
*                           // there's currently no way
*                           // of saying "stderr" in the
*                           // configuration language.
*                           // i.e. don't try this at home.
*         severity info; * };
*
*     default_stderr only works before the server daemonizes (i.e.
*     during initial startup) or when it is running in foreground
*     mode (-f command line option).
*/
/*
* There are many categories, so you can send the logs
* you want to see wherever you want, without seeing logs you
* don't want. Right now the categories are
*
*     default           the catch-all. many things still
*                       aren't classified into categories,
and
*                       they all end up here. also, if you
*                       don't specify any channels for a
used
*                       category, the default category is
*                       instead.
*     config           high-level configuration file
*                       processing
*     parser           low-level configuration file
processing
*     queries          what used to be called "query
logging"
*     lame-servers    messages like "Lame server on ..."
*     statistics
*     panic           if the server has to shut itself
*                       down due to an internal problem, it
*                       logs the problem here (as well as
*                       in the problem's native category)
*     update          dynamic update
*     ncache          negative caching
*     xfer-in         zone transfers we're receiving

```



```

*       xfer-out           zone transfers we're sending
*       db                 all database operations
*       eventlib           debugging info from the event system
*                           (see below)
*       packet             dumps of packets received and sent
*                           (see below)
*       notify             the NOTIFY protocol
*       cname              messages like "XX points to a CNAME"
*       security           approved/unapproved requests
*       os                 operating system problems
*       insist             consistency check failures
*       maintenance       periodic maintenance
*       load               zone loading
*       response-checks   messages like
*                           "Malformed response ..."
*                           "wrong ans. name ..."
*                           "unrelated additional info ..."
*                           "invalid RR type ..."
*                           "bad referral ..."
*/

category parser {
    syslog_errors;           // you can log to as many channels
    default_syslog;         // as you want
};

category lame-servers { null; }; // don't log these at all

channel moderate_debug {
    severity debug 3;       // level 3 debugging to file
    file "foo";            // foo
    print-time yes;        // timestamp log entries
    print-category yes;    // print category name
    print-severity yes;    // print severity level
    /*
     * Note that debugging must have been turned on either
     * on the command line or with a signal to get debugging
     * output (non-debugging output will still be written to
     * this channel).
     */
};

/*
 * If you don't want to see "zone XXXX loaded" messages but do
 * want to see any problems, you could do the following.
 */
channel no_info_messages {
    syslog;
    severity notice;
};

category load { no_info_messages; };

/*
 * You can also define category "default"; it gets used when no
 * "category" statement has been given for a category.
 */
category default {
    default_syslog;
    moderate_debug;
};

```

```
/*
 * If you don't define category default yourself, the default
 * default category will be used. It is
 *
 *     category default { default_syslog; default_debug; };
 */

/*
 * If you don't define category panic yourself, the default
 * panic category will be used. It is
 *
 *     category panic { default_syslog; default_stderr; };
 */

/*
 * Two categories, 'packet' and 'eventlib', are special. Only one
 * channel may be assigned to each of them, and it must be a
 * file channel. If you don't define them yourself, they default to
 *
 *     category eventlib { default_debug; };
 *
 *     category packet { default_debug; };
 */
};

include "filename"; // can't do within a statement
```

C BIND 8.1 Enhanced Features

The following points are explained in this appendix.

1. BIND 8 highlights
2. BIND Configuration File Guide — Logging Statement
3. BIND Configuration File Guide — Zone Statement
4. BIND Configuration File Guide — Option Statement
5. Converting From BIND 4.9.x

BIND 8 Highlights

- DNS Dynamic Updates (RFC 2136)
- DNS Change Notification (RFC 1996)
- Completely new configuration syntax
- Flexible, categorized logging system
- IP-address-based access control for queries, zone transfers, and updates that may be specified on a zone-by-zone basis
- More efficient zone transfers
- Improved performance for servers with thousands of zones
- The server no longer forks for outbound zone transfers
- Many bug fixes

BIND 8 is much more configurable than the previous release of BIND. There are entirely new areas of configuration, such as access control lists and categorized logging. Many options that previously applied to all zones can now be used selectively. These features, plus a consideration of future configuration needs led to the creation of a new configuration file format.

BIND Configuration File Guide — Logging Statement

Syntax

```
logging {  
    [ channel channel_name {
```

```

    ( file path_name
      [ versions ( number | unlimited ) ]
      [ size size_spec ]
      | syslog ( kern | user | mail | daemon | auth | syslog | lpr |
        news | uucp | cron | authpriv | ftp |
        local0 | local1 | local2 | local3 |
        local4 | local5 | local6 | local7 )
      | null );
  [ severity ( critical | error | warning | notice |
    info | debug [ level ] | dynamic ); ]
  [ print-category yes_or_no; ]
  [ print-severity yes_or_no; ]
  [ print-time yes_or_no; ]
}; ]

[ category category_name {
  channel_name; [ channel_name; ... ]
}; ]
...
};

```

Definition and Usage

The `logging` statement configures a wide variety of logging options for the nameserver. Its channel phrase associates output methods, format options and severity levels with a name that can then be used with the category phrase to select how various classes of messages are logged.

Only one logging statement is used to define as many channels and categories as are wanted. If there are multiple logging statements in a configuration, the first defined determines the logging, and warnings are issued for the others. If there is no logging statement, the logging configuration will be:

```

logging {
    category default { default_syslog; default_debug; };
    category panic { default_syslog; default_stderr; };
    category packet { default_debug; };
    category eventlib { default_debug; };
};

```

The Channel Phrase

All log output goes to one or more “channels”; make as many of them as you want.

Every channel definition must include a clause that says whether messages selected for the channel go to a file, to a particular syslog facility, or are discarded. It can optionally also limit the message severity level that will be accepted by the channel (default is “info”), and whether to include a named generated time stamp, the category name and/or severity level (default is not to include any).

The word `null` as the destination option for the channel will cause all messages sent to it to be discarded; other options for the channel are meaningless.

The file clause can include limitations both on how large the file is allowed to become, and how many versions of the file will be saved each time the file is opened.

The size option for files is simply a hard ceiling on log growth. If the file ever exceeds the

size, then `named` will just not write anything more to it until the file is reopened; exceeding the size does not automatically trigger a reopen. The default behavior is to not limit the size of the file.

If you use the `version logfile` option, then `named` will retain many backup versions of the file by renaming them when opening. For example, if you choose to keep 3 old versions of the file `lamers.log` then just before it is opened `lamers.log.1` is renamed to `lamers.log.2`, `lamers.log.0` is renamed to `lamers.log.1`, and `lamers.log` is renamed to `lamers.log.0`. No rolled versions are kept by default. The `unlimited` keyword is synonymous with `99` in current BIND releases.

The argument for the `syslog` clause is a syslog facility described earlier in this manual. How syslog will handle messages sent to this facility is described under `syslog.conf` earlier in this manual. If you have a system which uses a very old version of syslog and that only uses two arguments to the `openlog()` function, then this clause is silently ignored.

The severity clause works like syslog's "priorities", except that they can also be used if you are writing straight to a file rather than using syslog. Messages which are not at least of the severity level given will not be selected for the channel; messages of higher severity levels will be accepted.

If you are using syslog, then the `syslog.conf` priorities will also determine what eventually passes through. For example, defining a channel facility and severity as `daemon` and `debug` but only logging `daemon.warning` via `syslog.conf` will cause messages of severity information and notice to be dropped. If the situation were reversed, with `named` writing messages of only warning or higher, then syslog would print all messages it received from the channel.

The server can supply extensive debugging information when it is in debugging mode. If the server's global debug level is greater than zero, then debugging mode will be active. The global debug level is set either by starting the server with the `-d` flag followed by a positive integer, or by sending the server the `SIGUSR1` signal (for example, by using `ndc trace`). The global debug level can be set to zero, and debugging mode turned off, by sending the server the `SIGUSR2` signal (`ndc notrace`). All debugging messages in the server have a debug level, and higher debug levels give more detailed output. Channels that specify a specific debug severity, for example,

```
channel specific_debug_level {
    file "foo";
    severity debug 3;
};
```

will get debugging output of level 3 or less any time the server is in debugging mode, regardless of the global debugging level. Channels with dynamic severity use the server's global level to determine what messages to print.

If `print-time` has been turned on, then the date and time will be logged. `print-time` may be specified for a syslog channel, but is usually pointless since syslog also prints the date and time. If `print-category` is requested, then the category of the message will be logged as well. Finally, if `print-severity` is on, then the severity level of the message will be logged. The print options may be used in any combination, and will always be printed in the following order: time, category, and severity. Here is an example where all three print options are on:

```
28-Apr-1997 15:05:32.863 default: notice: Ready to answer queries.
```

There are four predefined channels that are used for named's default logging as follows. How they are used is described in the next section, The category phrase.

```
channel default_syslog {
    syslog daemon;          # send to syslog's daemon facility
    severity info;         # only send priority info and higher
};

channel default_debug {
    file "named.run";      # write to named.run in the working
directory
                           # Note: stderr is used instead of
"named.run"
                           # if the server is started with the "-f"
option.
severity dynamic; # log at the server's current debug level };

channel default_stderr {  # writes to stderr
    file "<stderr>";      # this is illustrative only; there's
currently
                           # no way of specifying an internal file
                           # descriptor in the configuration language.
severity info;           # only send priority info and higher
};

channel null {
    null;                  # toss anything sent to this channel
};
```

Once a channel is defined, it cannot be redefined. Thus you cannot alter the built-in channels directly, but you can modify the default logging by pointing categories at channels you have defined.

The Category Phrase

There are many categories, so you can send the logs you want to see wherever you want, without seeing logs you don't want. If you don't specify a list of channels for a category, then log messages in that category will be sent to the default category instead. If you don't specify a default category, the following "default" is used:

```
category default { default_syslog; default_debug; };
```

As an example, you want to log security events to a file, but you also want keep the default logging behavior. You'd specify the following:

```
channel my_security_channel {
    file "my_security_file";
    severity info
};
category security { my_security_channel; default_syslog; default_debug;
};
```

To discard all messages in a category, specify the null channel:

```
category lame-servers { null; };
category cname { null; };
```

The following categories are available:

default	The catch-all. Many things still aren't classified into categories, and they all end up here. Also, if you don't specify any channels for a category, the default category is used instead. If you do not define the default category, the following definition is used: <pre>category default { default_syslog; default_debug; };</pre>
config	High-level configuration file processing.
parser	Low-level configuration file processing.
queries	A short log message is generated for every query the server receives.
lame-servers	Messages like "Lame server on ..."
statistics	Statistics.
panic	If the server has to shut itself down due to an internal problem, it will log the problem in this category as well as in the problem's native category. If you do not define the panic category, the following definition is used: <pre>category panic { default_syslog; default_stderr; };</pre>
update	Dynamic updates.
ncache	Negative caching.
xfer-in	Zone transfers the server is receiving.
xfer-out	Zone transfers the server is sending.
db	All database operations.
eventlib	Debugging info from the event system. Only one channel may be specified for this category, and it must be a file channel. If you do not define the eventlib category, the following definition is used: <code>category eventlib { default_debug; };</code>
packet	Dumps of packets received and sent. Only one channel may be specified for this category, and it must be a file channel. If you do not define the packet category, the following definition is used: <code>category packet { default_debug; };</code>
notify	The NOTIFY protocol.
cname	Messages like "... points to a CNAME".
security	Approved/unapproved requests.
os	Operating system problems.
insist	Internal consistency check failures.
maintenance	Periodic maintenance events.
load	Zone loading messages.
response-checks	Messages arising from response checking, such as "Malformed response ...", "wrong ans. name ...", "unrelated additional info ...", "invalid RR type ...", and "bad referral ...".

BIND Configuration File Guide—Zone Statement

Syntax

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
    type master;
    file path_name;
    [ check-names ( warn | fail | ignore ); ]
    [ allow-update { address_match_list }; ]
    [ allow-query { address_match_list }; ]
    [ allow-transfer { address_match_list }; ]
    [ notify yes_or_no; ] [ also-notify { ip_addr; [ ip_addr; ... ] }; ]
};

zone domain_name [ ( in | hs | hesiod | chaos ) ]
{ type ( slave | stub );
  [ file path_name; ]
  masters { ip_addr; [ ip_addr; ... ] };
  [ check-names ( warn | fail | ignore ); ]
  [ allow-update { address_match_list }; ]
  [ allow-query { address_match_list }; ]
  [ allow-transfer { address_match_list }; ]
  [ max-transfer-time-in number; ]
  [ notify yes_or_no; ]
  [ also-notify { ip_addr; [ ip_addr; ... ] }; ]
};

zone "." [ ( in | hs | hesiod | chaos ) ] {
    type hint;
    file path_name;
    [ check-names ( warn | fail | ignore ); ]
};
```

Definition and Usage (Zone Types)

master	The master copy of the data in a zone.
slave	A slave zone is a replica of a master zone. The masters list specifies one or more IP addresses that the slave contacts to update its copy of the zone. If file is specified, then the replica will be written to the file. Use of file is recommended, since it often speeds server startup and eliminates a needless waste of bandwidth.
stub	A stub zone is like a slave zone, except that it replicates only the NS records of a master zone instead of the entire zone.
hint	The initial set of root nameservers is specified using a hint zone. When the server starts up, it uses the root hints to find a root nameserver and get the most recent list of root nameservers.

NOTE Previous releases of BIND used the term primary for a master zone, secondary for a slave zone, and cache for a hint zone.

Class

The zone's name may optionally be followed by a class. If a class is not specified, class in is

used.

Options

`check-names` See Name Checking.

`allow-query` See the description of `allow-query` in the Access Control section.

`allow-update` Specifies which hosts are allowed to submit Dynamic DNS updates to the server. The default is to deny updates from all hosts.

`allow-transfer` See the description of `allow-transfer` in the Access Control section.

`max-transfer-time-in` See the description of `max-transfer-time-in` in the Zone Transfers section.

`notify` See the description of `notify` in the Boolean Options section.

`also-notify` `also-notify` is only meaningful if `notify` is active for this zone.

The set of machines that will receive a DNS NOTIFY message for this zone is made up of all the listed nameservers for the zone (other than the primary master) plus any IP addresses specified with `also-notify`. `also-notify` is not meaningful for stub zones. The default is the empty list.

BIND Configuration File Guide — Options Statement

Syntax

```
options {
    [ directory path_name; ]
    [ named-xfer path_name; ]
    [ dump-file path_name; ]
    [ memstatistics-file path_name; ]
    [ pid-file path_name; ]
    [ statistics-file path_name; ]
    [ auth-nxdomain yes_or_no; ]
    [ deallocate-on-exit yes_or_no; ]
    [ fake-iquery yes_or_no; ]
    [ fetch-glue yes_or_no; ]
    [ host-statistics yes_or_no; ]
    [ multiple-cnames yes_or_no; ]
    [ notify yes_or_no; ]
    [ recursion yes_or_no; ]
    [ forward ( only | first ); ]
    [ forwarders { [ in_addr ; [ in_addr ; ... ] ] }; ]
    [ check-names ( master | slave | response ) ( warn | fail | ignore); ]
    [ allow-query { address_match_list }; ]
    [ allow-transfer { address_match_list }; ]
    [ listen-on [ port ip_port ] { address_match_list }; ]
    [ query-source [ address ( ip_addr | * ) ]
    [ port ( ip_port | * ) ]; ]
    [ max-transfer-time-in number; ]
    [ transfer-format ( one-answer | many-answers ); ]
    [ transfers-in number; ]
    [ transfers-out number; ]
    [ transfers-per-ns number; ]
}
```

```

[ coresize size_spec ; ]
[ datasize size_spec ; ]
[ files size_spec ; ]
[ stacksize size_spec ; ]
[ cleaning-interval number; ]
[ interface-interval number; ]
[ statistics-interval number; ]
[ topology { address_match_list }; ]
};

```

Definition and Use

The options statement sets up global options to be used by BIND. This statement may appear at only once in a configuration file; if more than one occurrence is found, the first occurrence determines the actual options used, and a warning will be generated. If there is no options statement, an options block with each option set to its default will be used.

Pathnames

- directory** The working directory of the server. Any non-absolute pathnames in the configuration file will be taken as relative to this directory. The default location for most server output files, for example, “named.run” is this directory. If a directory is not specified, the working directory defaults to “.”, the directory from which the server was started. The directory specified should be an absolute path.
- named-xfer** The pathname to the named-xfer program that the server uses for inbound zone transfers. If not specified, the default is system dependent for example, “/usr/sbin/named-xfer”.
- dump-file** The pathname of the file the server dumps the database to when it receives SIGINT signal (ndc dumpdb). If not specified, the default is “named_dump.db”.
- memstatistics- file** The pathname of the file the server writes memory usage statistics to on exit, if deallocate-on-exit is yes. If not specified, the default is “named.memstats”.
- pid-file** The pathname of the file the server writes its process ID in. If not specified, the default is operating system dependent, but is usually “/var/run/named.pid” or “/etc/named.pid”. The pid-file is used by programs like “ndc” that want to send signals to the running nameserver.
- statistics- file** The pathname of the file the server appends statistics to when it receives SIGILL signal (ndc stats). If not specified, the default is “named.stats”.

Boolean Options

auth-nxdomain If yes, then the AA bit is always set on NXDOMAIN responses, even if the server is not actually authoritative. The default is yes. Do not turn off auth-nxdomain unless you are sure you know what you are doing, as some older software won't like it.

deallocate-on- exit If yes, then when the server exits it will painstakingly

deallocate every object it allocated, and then write a memory usage report to the `memstatistics-file`. The default is `no`, because it is faster to let the operating system clean up. `deallocate-on-exit` is handy for detecting memory leaks.

- `fake-iquery` If yes, the server will simulate the obsolete DNS query type `IQUERY`. The default is `no`.
- `fetch-glue` If yes (the default), the server will fetch “glue” resource records it doesn’t have when constructing the additional data section of a response. `fetch-glue no` can be used in conjunction with `recursion no` to prevent the server’s cache from growing or becoming corrupted (at the cost of requiring more work from the client).
- `host- statistics` If yes, then statistics are kept for every host that the nameserver interacts with. The default is `no`.

NOTE Turning on `host-statistics` can consume huge amounts of memory.

- `multiple- cnames` If yes, then multiple `CNAME` resource records will be allowed for a domain name. The default is `no`. Allowing multiple `CNAME` records is against standards and is not recommended. Multiple `CNAME` support is available because previous versions of BIND allowed multiple `CNAME` records, and these records have been used for load balancing by a number of sites.
- `notify` If yes (the default), DNS `NOTIFY` messages are sent when a zone the server is authoritative for changes. The use of `NOTIFY` speeds convergence between the master and its slaves. Slave servers that receive a `NOTIFY` message and understand it, will contact the master server for the zone and see if they need to do a zone transfer, and if they do, they will initiate it immediately. The `notify` option may also be specified in the zone statement, in which case it overrides the options `notify` statement.
- `recursion` If yes, and a DNS query requests recursion, then the server will attempt to do all the work required to answer the query. If recursion is not on, the server will return a referral to the client if it doesn’t know the answer. The default is `yes`. See also `fetch-glue`.

Forwarding

The forwarding facility can be used to create a large sitewide cache on a few servers, reducing traffic over links to external nameservers. It can also be used to allow queries by servers that do not have direct access to the Internet, but wish to look up exterior names anyway. Forwarding occurs only on those queries for which the server is not authoritative and does not have the answer in its cache.

- `forward` This option is only meaningful if the forwarders list is not empty. A value of `first`, the default, causes the server to query the forwarders first, and if that doesn’t answer the question the server will then look for the answer itself. If only `is` is specified, the server will only query the forwarders.

`forwarders` Specifies the IP addresses to be used for forwarding. The default is the empty list (no forwarding).

Future versions of BIND 8 will provide a more powerful forwarding system. The syntax described above will continue to be supported.

Name Checking

The server can check domain names based upon their expected client contexts. For example, a domain name used as a hostname can be checked for compliance with the RFCs defining valid hostnames.

Three checking methods are available:

`ignore` No checking is done.

`warn` Names are checked against their expected client contexts. Invalid names are logged, but processing continues normally.

`fail` Names are checked against their expected client contexts. Invalid names are logged, and the offending data is rejected.

The server can check names in three areas; master zone files, slave zone files, and in responses to queries the server has initiated. If `check-names response fail` has been specified, and answering the client's question would require sending an invalid name to the client, the server will send a `REFUSED` response code to the client.

The defaults are:

```
check-names master fail;
check-names slave warn;
check-names response ignore;
```

`check-names` may also be specified in the zone statement, in which case it overrides the `options check-names` statement. When used in a zone statement, the area is not specified (because it can be deduced from the zone type).

Access Control

Access to the server can be restricted based on the IP address of the requesting system. See `address_match_list` for details on how to specify IP address lists.

`allow-query` Specifies which hosts are allowed to ask ordinary questions. `allow-query` may also be specified in the zone statement, in which case it overrides the `options allow-query` statement. If not specified, the default is to allow queries from all hosts.

`allow-transfer` Specifies which hosts are allowed to receive zone transfers from the server. `allow-transfer` may also be specified in the zone statement, in which case it overrides the `options allow-transfer` statement. If not specified, the default is to allow transfers from all hosts.

Interfaces

The interfaces and ports that the server will answer queries from may be specified using the `listen-on` option. `listen-on` takes an optional port, and an `address_match_list`.

The server will listen on all interfaces allowed by the address match list. If a port is not specified, port 53 will be used.

Multiple `listen-on` statements are allowed. For example:

```
listen-on { 5.6.7.8; };
listen-on port 1234 { !1.2.3.4; 1.2/16; };
```

If no `listen-on` is specified, the server will listen on port 53 on all interfaces.

Query Address

If the server doesn't know the answer to a question, it will query other nameservers. `query-source` specifies the address and port used for such queries. If address is `*` or is omitted, a wildcard IP address (`INADDR_ANY`) will be used. If port is `*` or is omitted, a random unprivileged port will be used. The default is

```
query-source address * port *;
```

NOTE `Query-source` currently applies only to UDP queries; TCP queries always use a wildcard IP address and a random unprivileged port.

Zone Transfers

`max-transfer-time-in` Inbound zone transfers (`named-xfer` processes) running longer than this many minutes will be terminated. The default is 120 minutes (2 hours).

`transfer-format` The server supports two zone transfer methods. `one-answer` uses one DNS message per resource record transferred. `many-answers` packs as many resource records as possible into a message. `many-answers` is more efficient, but is only known to be understood by BIND 8.1 and patched versions of BIND 4.9.5. The default is `one-answer`. `transfer-format` may be overridden on a `per-server` basis by using the `server` statement.

`transfers-in` The maximum number of inbound zone transfers that can be running concurrently. The default value is 10. Increasing `transfers-in` may speed up the convergence of slave zones, but it also may increase the load on the local system.

`transfers-out` This option will be used in the future to limit the number of concurrent outbound zone transfers. It is checked for syntax, but is otherwise ignored.

`transfers-per-ns` The maximum number of inbound zone transfers (`named-xfer` processes) that can be concurrently transferring from a given remote nameserver. The default value is 2. Increasing `transfers-per-ns` may speed up the convergence of slave zones, but it also may increase the load on the remote nameserver. `transfers-per-ns` may be overridden on a `per-server` basis by using the `transfers` phrase of the `server` statement.

Resource Limits

The server's usage of many system resources can be limited. Some operating systems don't support some of the limits. On such systems, a warning will be issued if the unsupported

limit is used. Some operating systems don't support limiting resources, and on these systems a cannot set resource limits on this system message will be logged.

Scaled values are allowed when specifying resource limits. For example, 1G can be used instead of 1073741824 to specify a limit of one gigabyte. unlimited requests unlimited use, or the maximum available amount. default uses the limit that was in force when the server was started. See `size_spec` for more details.

<code>coresize</code>	The maximum size of a core dump. The default is default.
<code>datasize</code>	The maximum amount of data memory the server may use. The default is default.
<code>files</code>	The maximum number of files the server may have open concurrently. The default is unlimited.

NOTE On some operating systems the server cannot set an unlimited value and cannot determine the maximum number of open files the kernel can support. On such systems, choosing unlimited will cause the server to use the larger of the `rlim_max` for `RLIMIT_NOFILE` and the value returned by `sysconf (_SC_OPEN_MAX)`. If the actual kernel limit is larger than this value, use `limit files` to specify the limit explicitly.

<code>stacksize</code>	The maximum amount of stack memory the server may use. The default is default.
------------------------	--

Periodic Task Intervals

`cleaning-interval` The server will remove expired resource records from the cache every `cleaning-interval` minutes. The default is 60 minutes. If set to 0, no periodic cleaning will occur.

`interface-interval` The server will scan the network interface list every `interface-interval` minutes. The default is 60 minutes. If set to 0, interface scanning will only occur when the configuration file is loaded. After the scan, listeners will be started on any new interfaces (provided they are allowed by the `listen-on` configuration). Listeners on interfaces that have gone away will be cleaned up.

`statistics-interval` Nameserver statistics will be logged every `statistics-interval` minutes. The default is 60. If set to 0, no statistics will be logged.

Topology

All other things being equal, when the server chooses a nameserver to query from a list of nameservers, it prefers the one that is topologically closest to itself. The `topology` statement takes an `address_match_list` and interprets it in a special way. Each top-level list element is assigned a distance. Non-negated elements get a distance based on their position in the list, where the closer the match is to the start of the list, the shorter the distance is between it and the server. A negated match will be assigned the maximum distance from the server. If there is no match, the address will get a distance which is

further than any non-negated list element, and closer than any negated element. For example,

```
topology {  
    10/8;  
    !1.2.3/24;  
    { 1.2/16; 3/8; };  
};
```

will prefer servers on network 10 the most, followed by hosts on network 1.2.0.0 (netmask 255.255.0.0) and network 3, with the exception of hosts on network 1.2.3 (netmask 255.255.255.0), which is preferred least of all.

The default topology is

```
topology { localhost; localnets; };
```

Converting From BIND 4.9.x

BIND 4.9.x configuration files can be converted to the new format by using `src/bin/named/named-bootconf.pl`, a perl script that is part of the BIND 8.1 source kit.

D Server Configuration Migration

There is a host of configuration migration utility available now. If you want to convert 4.x named.boot files to 8.x named.conf files, there is a perl script, named-bootconf.pl available on the system. This perl script file resides in /BIND/PUB/bin directory.

Explanation of configuration migration utilities;

The named-bootconf.pl is a perl script. Perl is a scripting language, like a shell script, it runs under an interpreter environment on MPE. The interpreter is a shareware, we require the Perl version 5 as the interpreter. The binary file for Perl version 5 can be downloaded from <http://jazz.external.hp.com>.

“Perl” is packaged as a mover archive. This has to be installed on the MPE machine. “Mover” is a archiving program which is available on <http://jazz.external.hp.com>. “Mover” will unarchive the “perl” package which is in mover formal and install at the correct place. One has to log on as MANAGER.SYS to do this. The files are stored in /usr/local/bin and /usr/local/lib/perl5 interpreter.

E Configure and Run Syslog/iX

How to Run Syslog/iX:

1. Log on as mgr.syslog.
2. Examine syslog.conf and customize for your own environment.
3. :stream JSYSLOGD.PUB.SYSLOG.
4. Stop Syslog/iX by issuing the command :ABORTJOB.##

```
##
## :TELL @.@
##
*.emerg      *
##
## Write to the :CONSOLE
##
*.alert      /dev/console
##
## :TELL @.SYSLOG
##
*.crit       @.SYSLOG
##
## :TELL MANAGER.SYS
##
*.err        MANAGER.SYS
##
## Forward to syslogd on another host via UDP
##
*.warning    @some.host.running.syslogd
##
## Write to the :CONSOLE
*.info       /dev/console
##
## Write to a file
##
*.debug      /tmp/syslog.log
```

The messages coming from a program are classified into **critical, informative, alert, error, emergency** etc. The syslog configuration file tells the syslog daemon how to post these messages. They could be sent to the console or to a log file, a printer, a message sent to an administrator or to another machine. SYSLOG uses UDP to send to another machine.

Explanation of parameters in syslog configuration file:

Syslog has a set of parameters that can be configured. Messages are classified into several levels. These messages can be directed to different outputs like console, logfile and so forth. They can also be sent to another machine which runs a syslog daemon.

They are classified as follows:

debug

info

error

critical

warning

alert

emergency

Now these messages could also be sent to a particular user by using the “tell” option followed by the user name.

They can also be sent to another machine by using “@machine name”.

A

address An identifier defined and used by a particular protocol and associated software to distinguish one node from another.

address resolution In NS networks, the mapping of node names to IP addresses and the mapping of IP addresses to subnet addresses. *See also probe protocol, ARP.*

alias A character string that is used as an alternate name for a protocol or a node.

ARP Address Resolution Protocol. ARP provides IP to LAN station address resolution for Ethernet nodes on a LAN.

ARPA Advanced Research Projects Agency.

ARPANET The computer network of the Advanced Research Projects Agency.

ASCII American National Standard Code for Information Interchange. A character set using 7-bit code used for information interchange among data processing and data communications systems. The American implementation of International Alphabet No. 5.

B

binary mode Data transfer scheme in which no special character processing is performed. All characters are considered to be data and are passed through with no control actions being taken.

bind A system call that assigns a specific name and unique address to a socket, turning a socket (which is one end-point of the connection) into an actual file. Binding allows servers to register well-known addresses with the system and each client to register a specific address for itself. *See also socket and well-known addresses.*

bootp Internet Boot Protocol (BOOTP) used to start, or boot, LAN devices such as routers, printers, X-terminals, and diskless workstations.

BOOTPTAB.NET.SYS The configuration file for the Bootstrap protocol daemon, `bootpd`, that contains client and relay information.

C

client A node on the internetwork that asks to use one of the Internet Services on the host. For example, a Telnet client is the process that uses Telnet protocol to establish a virtual terminal on your system.

D

daemon A process that either waits for the occurrence of an event or waits to perform some specified task on a periodic basis. Daemons are typically started once, on system startup, and they frequently start other processes to handle service requests. The Internet daemon `inetd` is a good example of such a process.

datagram A message consisting of content *and* all of the information needed to deliver the content between one system and another. Datagrams are sent using the User Datagram Protocol, or UDP. *See also* **UDP**.

datagram service A connectionless service that transmits messages, or datagrams, from one system to another. Because datagrams are transmitted without relying on a pre-established network connection (hence the term *connectionless*), each datagram must contain all the information required for its delivery. The protocol associated with datagram service is UDP, or User Datagram Protocol. *See also* **datagram, protocol, and UDP**.

DCE Data circuit-terminating equipment. The interfacing equipment required in order to interface to data terminal equipment (DTE) and its transmission circuit. Synonyms: data communications equipment, dataset.

domain name A name designated for a system in ARPANET standard format. This name can be used by other nodes on the network to access the host for which it is configured.

DTC Datacommunications and Terminal Controller. The DTC is a hardware device, configured as a node on a LAN, that enables asynchronous devices to access HP 3000 Series 900 computers. Terminals can either be directly connected to the DTC, or they can be remotely connected through a Packet Assembler Disassembler (PAD). The DTC can be configured with DTC/X.25 Network Access cards and DTC/X.25 Network Access software. A DTC/X.25 iX Network Link consists of two software modules: the X.25 iX System Access software (on the host) and the DTC/X.25 Network Access software (on the DTC).

DTC Telnet Access An HP product providing Telnet connections from HP 9000 and non-HP systems running ARPA standard Telnet services to the HP 3000. The solution includes a Telnet Access Card (TAC) that resides in the DTC 72MX or DTC 48 and provides protocol conversion between Telnet and Avesta Flow Control Protocol (AFCP). Equivalent functionality is provided by a separate product, the Telnet Express Box (TEB).

DTE Data Terminal Equipment. Equipment that converts user information into data transmission signals or reconverts received data signals into user information. Data terminal equipment operations in conjunction with data circuit-terminating equipment.

E

environment A session that is established on a remote node.

Ethernet A Local Area Network system that uses baseband transmission at 10 Mbps over coaxial cable. Ethernet is a trademark of Xerox Corporation.

F

file equation An assignment statement that is used to associate a file with a specific device or type of device during execution of a program.

flow control A means of regulating the rate at which data transfer takes place between devices to protect against data overruns.

FTP File Transfer Protocol. The Internet Services protocol that facilitates the transfer of files between systems. Originally developed by the Advanced Research Projects Agency (ARPA).

H

host computer A computer on which network communications software resides, and which is currently providing a service to a requesting client.

HOSTS.NET.SYS The host name data base file which associates Internet addresses with official host names and aliases.

I

IEEE 802.3 A standard for a broadcast local area network published by the Institute for Electrical and Electronics Engineers (IEEE). This standard is used for both the ThinLAN and ThickLAN implementations of the Local Area Network (LAN).

inetd The Internet server that allows one daemon to invoke many servers, thus reducing load on the system. Normally started at system boot time, only one `inetd` can run at any given time.

INETDCNF.NET.SYS The configuration file for the Internet daemon `inetd`, which determines which installed Internet Services are available to users.

INETDSEC.NET.SYS The optional security file for `inetd`, which lets you control access to individual services to specific accounts, groups, or users.

internet An aggregation of computer systems and other types of computing equipment that share information according to a set of defined communications protocols. Local networks, such as all computer systems linked together within a company, are typically linked to other local networks via the Internet. Or, individual systems which are not part of a local network, such as a personal computer or a standalone business computing system, can exchange information via the Internet if they are equipped with the appropriate communications software and hardware.

Internet Protocol (IP). A set of rules used to route information between different local networks in an internetwork, as well as among nodes in the same local network. The internet protocol corresponds to layer three, the network layer, of the OSI model. *See also IP address.*

IP address Internet Protocol address. An address used by the Internet Protocol to route information. A complete IP address comprises a network portion and a subnet portion to identify a specific network, and a node portion to identify a node within that network.

L

local host The host system you are currently working from.

local node Same as host system.

loopback The routing of messages from a node back to itself.

N

name space The set of possible names allowed in a given environment. The POSIX name space, which follows hierarchical file system syntax (i.e., `\sys\pub\myfile`) is distinct from the MPE/iX name space, which follows MPE naming rules (i.e., `MYFILE.PUB.SYS`).

network address Either the network portion of an IP address (as opposed to the node portion) or a node's X.25 address when referring to X.25 networks.

network directory A file containing information required for one node to communicate with other nodes in 1) an internetwork, 2) an X.25 network, or 3) a network that contains non-HP nodes. The active network directory on a node must be named `NSDIR.NET.SYS`.

NI *See Network Interface.*

Network Interface The collection of software that enables data communication between a system and a network. A node possesses one or more network interfaces for each of the networks to which it belongs. Examples of network interfaces include Local Area Networks (LANs), point-to-point (router), X.25, token ring, SNA, loopback, and gateway half. The maximum number of supportable network interfaces is 12, one of which is reserved for loopback.

Network Services Software application products that can be used to access data, initiate processes, and exchange information among nodes in the network. The NS 3000/iX Network Services include RPM, VT, RFA, RDBA, and NFT.

NMCONFIG.PUB.SYS The file that contains all the network configuration data for the HP 3000 Series 900 computer on which it resides. It includes information about the clients that can access the system as well as information about any Network Services (NS) products running on the system. This is the only file name allowed.

NMMAINT Node Management services MAINTenance utility. A utility that lists the software module version numbers for all HP AdvanceNet products, including NS 3000/iX. It detects missing or invalid software modules.

NMMGR Node Management Services Configuration Manager. A software subsystem that enables you to configure network connectivity and access parameters for an HP 3000 Series 900 computer.

NMMGRVER Node management services conversion utility. A conversion program that converts configuration files created with NMMGR from an earlier version to the latest format.

node A computer that is part of a network. The DTC, or Datacommunications and Terminal Controller that enables asynchronous devices to access the HP 3000, is also considered to be a node and has its own address.

node address The node portion of an Internet Protocol (IP) address.

Node Management Services Configuration Manager See NMMGR.

node name A character string that uniquely identifies each system in a network or internetwork. Each node name in a network or internetwork must be unique; however, a single node can be identified by more than one node name.

NS See Network Services.

NS 3000/iX Link Software and hardware that provides the connection between nodes on a network. Some of the NS 3000/iX links available are the ThinLAN 3000/iX Link and its ThickLAN option, the DTC/X.25 iX Network Link, and the NS Point-to-Point 3000/iX Link.

NS 3000/iX Network Services Software applications that can be used to access data, initiate processes, and exchange information among nodes in a network. The services are RPM, VT, RFA, RDBA, and NFT.

NSDIR.NET.SYS The name of the active network directory file. See also network directory.

P

packets Encapsulated messages transmitted across a network or an internetwork.

privileged mode A capability assigned to accounts, groups, or users allowing unrestricted memory access, access to privileged CPU instructions, and the ability to call privileged procedures.

probe protocol An HP protocol used by NS 3000/iX IEEE 802.3 networks to get information about other nodes on the network. It resolves names to IP addresses, and resolves IP addresses to IEEE 802.3 addresses.

process A single instance of a program that is being executed by the operating system, also known as a task.

protocol A set of rules that enables two or more data processing entities to exchange information. In networks, protocols are the rules and conventions that govern each layer of network architecture. They define what functions are performed and how messages are exchanged.

protocols file A file that contains a list of protocols known to the system, plus the identification number and one or more aliases for each. *See also* **protocol**.

PROTOCOL.NET.SYS The protocols file, described above.

R

relay Using one node on an internetwork to pass information through to another node or nodes. A relay entry in the `bootpd` configuration file, for example, provides the information necessary to forward, or relay, bootstrap protocol requests to one or more bootp servers.

remote host The host system from which you, as a client, are requesting service.

remote node A node on an internetwork other than the node you are currently using or referring to.

RSLVSAMP.NET.SYS Sample initialization file for the domain name resolver.

RESLVCNF.NET.SYS An initialization file for the domain name resolver. It contains information needed by the network to determine how to resolve a domain name to an IP address.

S

server A node on a network or internetwork that provides on-demand service to requesting clients.

services file The file which associates official service names and aliases with the port number and protocol the services use. In the HFS name space, this file is `/etc/services`.

SERVICES.NET.SYS The services name file, described above.

socket A special kind of file that uniquely identifies one end point of an Internetwork connection. A socket specifies the protocol being used (for example TCP) the Internetwork address (for example `192.44.244.7`) and the integer identifying the process (for example `377`). A socket pair completely specifies the two processes that make up an Internetwork connection.

stream services A type of service that uses Transmission Control Protocol (TCP) to exchange information on an internetwork. Stream services rely on an established, known connection between two systems, client and host, similar to a leased or dedicated phone line between two parties.

stream socket A type of socket that is used to establish stream services between two systems.

subnet Another name for a network, especially if the network is part of an internetwork. The word subnet is also a synonym for intranet.

subnet mask A grouping of bits that determines which bits of the IP address will be used to define a subnetwork. The subnet mask is configured using the NMMGR utility and specified in the same format as an IP address.

T

TAC Telnet Access Card. A board within a DTC 48 or 72MX.

TCP/IP Transmission Control Protocol/Internet Protocol. A set of rules that establishes and maintains connections between nodes on an internetwork. TCP/IP regulates the flow of data, breaks messages into smaller fragments if necessary (and reassembles the fragments at the destination), detects errors, and retransmits messages if errors have been detected.

TEB Telnet Express Box. An HP product consisting of a DTC dedicated to providing protocol conversion between Telnet on TCP/IP and AFCP to allow incoming calls from the Internet Services environment to HP 3000 systems.

Telnet The application protocol offering virtual terminal service in the Internet suite of protocols developed by the Advanced Research Projects Agency (ARPA).

TELNET.ARPA.SYS A file that contains the Telnet client program.

TELNTDOC.ARPA.SYS The readme file for the Telnet client program.

TFTP Trivial File Transfer Protocol, TFTP, a set of rules used to read and write files to or from a remote system.

U

UDP User Datagram Protocol, a set of rules used to send connectionless messages called *datagrams* between systems. UDP requires much less overhead than a protocol such as TCP because it does not require acknowledgement from the recipient that the message reached its destination.

V

Virtual Terminal A network service that allows a user to establish interactive sessions on a node.

W

WAN Wide Area Network. A data communications network of unlimited size, used for connecting localities, cities, and countries.

well-known address The port number that identifies the specific user process of an available and commonly-used Internet Service. For example, the port number for the File Transfer Protocol (FTP) is 21.

X

X.25 Defines the interface between a DTE and a DCE for packet mode operation on a public data network (PDN).

Symbols

\$STDLIST messages, 36
/etc directory, 19, 27
/etc/bootpd, 48
/etc/bootpquery, 48
/etc/bootptab, 20, 48, 50
/etc/hosts.equiv, 67
/etc/inetd, 26
/etc/inetd.conf, 19, 26, 27, 42, 49, 60
/etc/protocols, 19
/etc/services, 67
/usr/adm/inetd.sec, 19, 26

A

aborting JINETD, 34
access to the inetd security file, 30
adding Internet services, 27
additional information on Internet Services, 24
addr, 116
allow hosts, 83
allowing Telnet access, 46
Apache
 components, 124
 configuring, 127
 documentation, 132
 error logging, 130
 features, 122
 starting, 125
 stopping, 129
 verify activity, 128
Apache for MPE/iX, 121
ARPA.SYS, group capabilities, 41
Available, 84

B

ba tag, 52
Berkeley Internet Name Domain
 BIND, 101
bf tag, 52
BIND, 101
boot request
 listening for, 50
 packet, 56
 relaying, 51
bootfile home directory, 52
bootfile name, 52
bootfile size, 52
booting a printer, 48
booting devices, 47
bootpd
 address list, 53

 boot file name, 51
 bootfile name, 52
 bootfile size, 52
 broadcast address, 52
 client data, 51
 client's gateway address, 51
 client's hardware address, 51
 client's IP address, 51
 client's system name, 51
 command line options, 55
 configuration file entries, 51
 configuration file tags, 52
 configuring, 48
 files, 48
 gateway addresses, 52
 hardware addresses, 52
 hardware type code, 52
 home directory, 52
 implementation differences, 57
 magic cookie, 53
 maximum hops, 51
 network interface hardware, 51
 operation, 55
 overview, 48
 problems with, 56
 relay data, 51
 RFC1034 server addresses, 52
 specifying the configuration file, 55
 starting, 55
 threshold value, 51
 time zone offset, 53
 timeout value, 55
 timing out, 55
 vendor field, 53
bootpd configuration file, 20
 creating, 50
 editing tips, 53
 linking, 50
 sample, 54
bootpd logging messages, 55
BOOTPD.NET.SYS, 48
BOOTPQRY
 diagnostic options, 56
 operation, 56
BOOTPQRY.NET.SYS, 48
BOOTPTAB.NET.SYS, 20, 50
BPTABSMP.NET.SYS, 20, 48, 50
broadcast address, 52
browseable, 84
bs tag, 52

C

- c command, 34, 42, 49, 61, 62
- capabilities
 - for ARPA.SYS, 41
 - for MGR.TFTP, 61
 - for USER.TFTP, 61
- chargen service, 26
- client data in bootpd, 51
- command
 - negprot, 78
 - sesssetup, 78
 - TconX, 79
- command options for inetd, 34
- comment, 84
- config file, 80
- configuration files
 - bootpd, 20, 55
 - copying, 18
 - inetd, 19, 26, 27
 - Internet daemon, 27
 - linking, 19
 - required, 19
 - samples, 18
- configuring
 - BOOTP server, 48, 57
 - Internet Services, 18
 - remsh client, 66
 - Telnet server, 42
 - TFTP server, 60
- connection attempts, message syntax, 35
- connection logging
 - turning off, 35
 - turning on, 35
- console messages, 33, 36
- controlling access to the Internet Services, 30
- copying the bootpd sample file, 50
- copying the configuration files, 18
- copying the inetd sample file, 27
- create mode, 85
- creating
 - TFTP account, 61
 - TFTPDIR, 61
- creating symbolic links, 19, 27, 50
- creating the inetd security file, 30
- creating the protocols file, 23
- creating the services file, 20

D

- daemon, 16
- daemon, Internet, 16
- datagram

- multi-threaded, 29
- services, 25
- single-threaded, 29
- sockets, 25
- datagram services, 25
- datagram sockets, 26
- daytime service, 26
- dead time, 81
- debug level, 82
- default service, 80
- deny hosts, 83
- denying Telnet access, 46
- devices, booting, 47
- dig, 103, 115
- discard service, 26
- DNS, 117
 - change notification, 102
 - dynamic updates, 102
- dnsquery, 116
- Domain Name System, 117
 - DNS, 102
- domain names, resolving, 18
- ds tag, 52
- DTC Telnet access, 39

E

- echo service, 26
- editing
 - bootpd configuration file, 53
 - inetd configuration file, 27
 - inetd security file, 30
 - services file, 21, 42, 48, 60
- entries in the bootpd configuration file, 51
- entries in the inetd configuration file, 29
- error messages
 - inetd, 33, 36
 - tftpd, 63
- event logging for inetd, 36

F

- fields in the inetd configuration file, 29
- file access
 - for TFTP clients, 61
 - to /etc/bootptab, 50
 - to BOOTPTAB.NET.SYS, 50
 - with tftpd, 62
- file copying
 - bootpd configuration file, 50
 - inetd configuration file, 27
 - inetd security file, 30
 - protocols file, 23

- services file, 20
- file creation
 - bootpd configuration file, 50
 - inetd configuration file, 27
 - inetd security file, 30
 - protocols file, 23
 - services file, 20
- file linking, 19
 - bootpd configuration file, 50
 - inetd configuration file, 27
 - inetd security file, 30
 - protocols file, 23
 - services file, 20

G

- gateway addresses, 52
- guest account, 83
- guest ok, 84
- guest only, 85
- gw tag, 52

H

- ha tag, 52
- hardware addresses, 52
- hd tag, 52
- home directory of bootfile, 52
- ht tag, 52

I

- implementation differences
 - bootpd, 57
 - inetd, 38
 - remsh, 70
 - Telnet server, 46
 - tftpd, 63
- INCNFSMP.NET.SYS, 19, 26
- inetd
 - command options, 34
 - internal services, 26
 - overview of, 25
 - starting, 32
- inetd configuration
 - adding BOOTP service, 49
 - adding Telnet service, 42
 - adding TFTP service, 60
 - changing, 27
 - file creation, 27
 - file linking, 27
- inetd configuration file, 19, 26
 - editing, 27
 - editing tips, 29

- example, 28
- fields, 29
- linking, 27
- reading, 27
- rereading, 28
- inetd connection logging, 35
- inetd event logging, 36
- inetd job file, 26
- inetd messages, 36
- inetd operation, problems with, 36
- inetd program file, 26
- inetd security file, 19, 26, 30
 - creating, 30
 - editing, 30
 - editing tips, 31
 - example, 30
 - linking, 30
 - problems with, 37
 - range indicator in, 32
 - wildcard characters in, 32
 - write access to, 30
- INETD.NET.SYS, 26
- INETDCNF.NET.SYS, 19
- INETDSEC.NET.SYS, 19
- INSECSMP.NET.SYS, 19, 26
- installing configuration files, 19
- INTEDCNF.NET.SYS, 42, 49, 60
- interfaces, 81
- internal inetd services, 26
- internet, 15
- Internet daemon, 16
 - configuration file, 19
 - error reporting, 33
 - implementation differences, 38
 - logging connection attempts, 35
 - logging events, 36
 - security file problems, 37
 - status messages, 33
 - stopping, 34
 - troubleshooting, 36
- internet daemon, 25
- Internet daemon explained, 25
- Internet server, 16
- Internet Services
 - adding, 27
 - additional reading, 24
 - configuration files, 18
 - configuring, 18
 - controlling access to, 30
 - file linking, 19
 - overview, 15
 - required files, 17

requirements, 17
summarized, 15
table of, 15
updating, 28
verifying installation of, 17
invalid users, 83
ip tag, 53

J

JINETD
 aborting, 34
 changing the security, 33
 job number, 34
 password protection on, 33
JINETD.NET.SYS, 26, 32
job file, inetd, 26
job number for JINETD, 34
job passwords on JINETD.NET.SYS, 33

K

-k command, 34
killing inetd, 34
known protocols, 19

L

-l command, 34, 35
LAN devices, 47
linking
 bootpd configuration file, 50
 configuration files, 19
 inetd configuration file, 27
 inetd security file, 30
 protocols file, 23
 services file, 20
list of Internet Services, 15
load printers, 82
log file, 81
logging connection attempts in inetd, 35
logging events in inetd, 35
logging messages in bootpd, 55

M

master server, 25
master zone, 107
max log size, 81
max xmit, 81
message syntax, connection attempts, 35
MPE/iX examples, 68
multi-threaded datagram, 29

N

named.conf, 106
negprot command, 78
Net Transport software, 17
NET.SYS, 18
network interface link cards, 17
new inetd services, adding, 27
NMBD, 76
NMMAINT utility, 17, 41
nodes, 25
nowait state, 29
nslookup, 103, 113, 115

O

option
 comment, 84
overview of Internet Services, 15

P

packets, 50
path, 84
permissions in the inetd security file, 30
POSIX examples, 69
POSIX file names, 19, 27
postexec, 82
preexec, 82
print command, 82
printcap name, 82
protocol
 bootstrap, 48, 60
 TFTP, 59
 UDP, 59
protocol field, 29
PROTOCOL.NET.SYS, 19
protocols, 15
protocols file, 19, 23
 creating, 23
 linking, 23
 viewing, 23
PROTSAMP.NET.SYS, 19

R

read only, 85
reading the inetd configuration file, 27
recommended books, 24
reconfiguring bootpd, 50
relay data in bootpd, 51
remote shell, 65
remsh service, 65
required configuration files, 19

-
- rereading the inetd configuration file, 28, 34, 42, 49, 61, 62
 - Resolvers, 104
 - resolving domain names, 18
 - restarting inetd, 32
 - RFC1034 server addresses, 52

 - S**
 - Samba/iX, 73
 - sample configuration files, 18, 27, 50
 - security
 - problems with, 37
 - security file
 - inetd, 19, 26
 - security permissions, 32
 - server, 16
 - Server Message Block (SMB), 74
 - server program field, 29
 - server string, 80
 - server, Internet, 16
 - service descriptions, 29
 - service name field, 29
 - service permissions, 30
 - services, 15
 - services file, 20
 - adding BOOTP service, 48
 - adding Telnet service, 42
 - adding TFTP service, 60
 - creating, 20
 - editing, 21
 - editing tips, 21
 - example of, 21
 - linking, 20
 - SIGHUP signal, 50
 - single-threaded datagram, 29
 - slave zone, 107
 - SMB, 74
 - Server Message Block, 74
 - SMB protocol, 76
 - SMB.CONF, 76
 - SMBCLIENT, 76
 - SMBD, 76
 - SOA, 109
 - socket, 25
 - socket services, 26
 - socket type field, 29
 - sources, recommended, 24
 - Start of Authority, 109
 - starting and stopping connection logging, 34
 - starting bootpd, 55
 - starting inetd, 32
 - automatically, 33
 - interactively, 33
 - manually, 32
 - starting tftpd, 62
 - stopping inetd, 34
 - stream, 25
 - stream services, 25
 - stream sockets, 25
 - streaming JINETD, 32, 33, 42, 49
 - summary of Internet Services, 15
 - superserver, 25
 - symbolic link, creating, 19, 27, 50
 - symbolic links and tftpd, 62
 - Syslog, 104
 - Syslog/iX, 104
 - syslogd server, 38
 - SYSSTART file, 33

 - T**
 - T### tag, 53
 - tags in bootpd configuration file, 52
 - TCP/IP, 25
 - TCP/IP protocol, 39
 - TCP/UDP/IP, 17
 - Telnet
 - file permissions, 41
 - operation, 40
 - verifying file installation, 41
 - Telnet Access Card (TAC), 39
 - Telnet access, using a DTC, 39
 - Telnet client
 - overview, 39
 - Telnet errors, 44
 - Telnet information, online, 39
 - Telnet protocols, 39
 - Telnet server
 - as an internal program, 46
 - configuring, 42
 - implementation differences, 46
 - overview, 39
 - security checking on, 46
 - telnetd server file, 46
 - troubleshooting, 44
 - TELNET.ARPA.SYS, 40
 - testing network devices for bootpd, 56
 - TESTPARM, 77
 - TESTPRNS, 77
 - TFTP account
 - creating, 61
 - creating the manager, 61
 - TFTP home directory, 61
 - TFTP user, 61
 - tftpd

- client's directory access, 61
- configuring, 60
- configuring the user, 61
- file access permission, 62
- file lookups, 61
- following symbolic links, 62
- implementation differences, 63
- operation, 60
- problems with, 63
- starting, 62
- the TFTP user, 63
- time service, 26
- to=offset tag, 53
- transmitting files with tftpd, 62
- Trivial File Transfer Protocol, 59
- trivial services, inetd, 26

U

- UDP/IP, 26
- updating the Internet Services, 28
- User Datagram Protocol, 59
- user field, 29
- USER.TFTP, 63

V

- valid users, 83

W

- wait state, 29
- wait state field, 29
- wildcard characters
 - in inetd security file, 32
- workgroup, 80
- wrappers, 46
- write ok, 85