Installing and Administering NFS Services with 10.20 ACE and HWE

HP 9000 Networking

B1031-90043

E0498

Printed in: U.S.A.

Legal Notices

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

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

Warranty. A copy of the specific warranty terms applicable to your Hewlett- Packard product and replacement parts can be obtained from your local Sales and Service Office.

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 for DOD agencies, and subparagraphs (c) (1) and (c) (2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies.

Hewlett-Packard Co.
19420 Homestead Road
Cupertino, CA 95014 USA

Use of this manual and flexible disk(s) or tape cartridge(s) supplied for this pack is restricted to this product only. Additional copies of the programs may be made for security and back-up purposes only. Resale of the programs in their present form or with alterations, is expressly prohibited.

Copyright Notices

Trademark Notices

UNIX is a registered trademark of the Open Group.

NFS® is a registered trademark of Sun Microsystems, Inc.

NIS™ is a trademark of Sun Microsystems, Inc.

NOTE

The Network Information Service (NIS) was formerly known as Yellow Pages (YP). The functionality is the same; only the name has changed. “Yellow Pages” is a registered trademark in the United Kingdom of British Telecommunications plc.

Contents

Contents

Contents

Contents

Contents

Contents

1 Overview of the NFS Services

This manual documents the HP-UX 10.20 version of the NFS Services, with the 10.20 ACE (Additional Core Enhancements) or HWE (Hardware Enhancements) installed.

This manual does not document NFS Diskless. For information on NFS Diskless configuration and administration, see the *Managing Systems and Workgroups* manual.

For more information, see *Managing NFS and NIS*, by Hal Stern, published by O'Reilly & Associates.

The 10.20 ACE and HWE

With the 10.20 ACE and HWE, the NFS Services include the following, which were not part of the original 10.20 NFS Services:

- **AutoFS**, the next generation of the NFS automounter. AutoFS solves many of the problems with the automounter. The syntax of the automounter maps does not change with AutoFS, but some of the command-line options are different. See "To Migrate from the Automounter to AutoFS" on page 53.
- **CacheFS**, a local file system type for caching information that is NFS-mounted from a remote server. CacheFS improves read performance for information that is read repeatedly. See Chapter 3, "Configuring the Cache File System (CacheFS)," on page 95.
- **NFS Protocol Version 3 (NFS PV3)**, the next version of NFS, which improves NFS performance and supports larger files. By default, the local NFS client will attempt to mount a file system using NFS version 3. If the NFS server does not support version 3, the file system will be mounted using version 2. You can specify the mount option `vers=2` to force NFS to use NFS PV2. See "To Change the Default Mount Options" on page 40.

The Compatibility Switch

The 10.20 ACE and HWE include a system-wide compatibility switch to control the behavior of certain file system APIs.The behavior of the `stat()`, `statfs()`, and `statvfs()` functions is affected. The `fstat()`, `fstatfs()`, and `fstatvfs()` functions are affected as well, by virtue of the fact that they use the `stat()`, `statfs()`, and `statvfs()` functions.

If the switch is in compatibility mode (compatible with the original 10.20 system behavior, which is the default), return values from the `stat()`, `statfs()`, and `statvfs()` system calls are unaffected. With the switch in non-compatibility mode, these calls return different values in the `st_fstype` field of the `stat` structure returned by `stat()`, the `f_fsid` field of the `statfs` structure returned by `statfs()`, or the `f_fsid` or `f_fsindex` field of the `statvfs` structure returned by `statvfs()`.

The values returned are appropriate to the type of file system being queried. Calls to `sysfs()` with these values will return `nfs` for NFS Version 2 file systems, `nfs3` for NFS Version 3 file systems, `autofs` for unmounted file systems being monitored by AutoFS, and `cachefs` for CacheFS mounts. (CacheFS file systems normally return the value of the underlying mount, except for the `f_basetype` value in the `statvfs` structure, which will contain the value `cachefs` for the CacheFS file system.)

To set the switch to compatibility mode (the default), type the following:

```
onccompat -c
```

To set the switch to non-compatibility mode, type the following:

```
onccompat -n
```

For more information, see the `onccompat`(1M) man page.

The compatibility switch is available only on the 10.20 ACE and HWE. HP-UX 10.30 and 11.0 already implement the non-compatible behavior.

The NFS Services

Hewlett-Packard's NFS Services include the following:

- **Network File System (NFS)** provides transparent access to files from anywhere on the network. An NFS server makes a directory available to other hosts on the network by "exporting" the directory. An NFS client provides access to the NFS server's directory by "mounting" the directory. To users on the NFS client, the directory looks like part of the local file system. For information on configuring and administering NFS, see "Configuring and Administering NFS" on page 17.
- **Network Information Service (NIS)** allows centralized management of common configuration files, like `/etc/passwd`, `/etc/hosts`, and `/etc/services`. An NIS "master server" holds master copies of the configuration files, or "maps". The master server may distribute copies of the maps to NIS "slaves servers" to provide load balancing and reliability. An NIS client gets configuration information from the master server or a slave server instead of from its local configuration files. (Some local configuration files, like `/etc/passwd` and `/etc/group`, can be used in addition to the NIS maps.) For more information, see "Configuring and Administering NIS" on page 101.
- **Network Lock Manager and Network Status Monitor (rpc.lockd and rpc.statd)** provide file locking and synchronized file access to files that are shared with NFS. Files may be locked with `lockf` or `fcntl`. For more information, see the following man pages: `lockd`(1M), `statd`(1M), `lockf`(2), and `fcntl`(2).
- **Remote Procedure Call (RPC)** is the mechanism that allows NFS clients and NFS servers to communicate. You can write your own RPC applications, using **rpcgen**, an RPC compiler that simplifies RPC programming. On HP-UX 10.30 and later, Transport-Independent RPC (TI-RPC) is supported. For information on RPC and `rpcgen`, see *Power Programming with RPC*, by John Bloomer, published by O'Reilly and Associates, Inc.
- **Remote Execution Facility (REX)** allows you to execute commands interactively on a remote host while your local environment is simulated on the remote host. To use REX, you issue the `on` command on your local host, supplying the command you want

to execute remotely and the name of the remote host where you want the command to execute. Your current environment variables are then copied to the remote host, and your home directory is mounted on the remote host using NFS. For information on configuring, administering, and using REX, see "Configuring and Using the Remote Execution Facility (REX)" on page 165.

- The `rup` command collects and displays status information about the hosts on the local network. All hosts running the `rstatd` daemon will respond to queries from the `rup` command. For more information, see the man pages `rstatd`(1M) and `rup`(1). For information on configuring `rstatd`, see "Configuring the Other NFS Daemons and Services" on page 91.
- The `rusers` command collects and displays information about all users logged into the hosts on the local network. All hosts running the `rusersd` daemon will respond to queries from the `rusers` command. For more information, see the man pages `rusersd`(1M) and `rusers`(1). For information on configuring `rusersd`, see "Configuring the Other NFS Daemons and Services" on page 91.
- The `rwall` program allows you to broadcast a message to all the users logged into a remote host. The `rwall` program sends a message to a specified host where the `rwalld` daemon is running. The `rwalld` daemon then writes the message to all the users logged into that host. For more information, see the man pages `rwalld`(1M) and `rwall`(1M). For information on configuring `rwalld`, see "Configuring the Other NFS Daemons and Services" on page 91.
- The `spray` command sends a stream of packets to a specified host and then reports how many of the packets were received and what the transfer rate was. All hosts running the `sprayd` daemon will repond to packets sent by the `spray` command. For more information, see the man pages `sprayd`(1M) and `spray`(1M). For information on configuring `sprayd`, see "Configuring the Other NFS Daemons and Services" on page 91.
- The `quota` command, which displays information about a user's disk usage and limits, may be used to get information about a user on a remote host, if the `rquotad` daemon is running on the remote host. For more information, see the man pages `rquotad`(1M) and `quota`(1). For information on configuring `rquotad`, see "Configuring the Other NFS Daemons and Services" on page 91.

2 Configuring and Administering NFS

This chapter tells you how to configure and administer an HP 9000 as an NFS server or client, by editing files and issuing HP-UX commands.

An **NFS server** is a machine that "exports" (makes available) its local files and directories to NFS clients. An **NFS client** is a machine that "mounts" files and directories exported by NFS servers. NFS-mounted files and directories look to users like part of the NFS client's local file system.

A machine can be an NFS server and an NFS client at the same time.

NOTE

HP does not support NFS or NIS over Wide Area Networks (WANs). WANs include network links using X.25, microwave links, public common carriers, or high speed lines (such as 56kb).

HP offers limited support of NFS over extended LANs "Limited support" means that HP cannot unilaterally support every conceivable extended LAN topology for NFS, but HP will support LAN configurations on local LAN media between HP servers. These include 802.3 and FDDI segments separated by routers or bridges. Network Support *must* be purchased by the customer for support of NFS over extended LANs.

This chapter is intended for system administrators who prefer not to use SAM. However, Hewlett-Packard recommends that you use SAM to configure and administer NFS. SAM (System Administration Manager) is Hewlett-Packard's windows-based user interface for performing system administration tasks. To run SAM, type `sam` at the HP-UX prompt. SAM has an extensive online help facility.

This chapter contains the following sections:

- Preparing for NFS Configuration
- Configuring and Administering an NFS Server
- Configuring and Administering an NFS Client
- Configuring and Administering AutoFS
- Configuring and Using NFS Netgroups
- Configuring the Other NFS Daemons and Services

Preparing for NFS Configuration

Before you configure your machine as an NFS server or client, you must perform the following tasks:

1. To Check the Network Connections
2. To Set User IDs and Group IDs (if NIS is not used)
3. To Ensure that No User is a Member of Too Many Groups

The rest of this section explains the procedures for performing these tasks.

To Check the Network Connections

- Issue the `/usr/sbin/ping`(1M) command for each system with which your system will communicate using NFS.

  If the `ping`(1M) command fails, see the manuals listed below for troubleshooting procedures.

Before you configure NFS, you must have already installed and configured the network hardware and software on all the machines that will use NFS. For information on installing and configuring the network hardware and software, refer to the following manuals:

Installing and Administering LAN/9000 Software

Installing and Administering Token Ring/9000 Software

Installing and Administering FDDI/9000 Software

To Set User IDs and Group IDs (if NIS is not used)

- Create one `/etc/passwd` file and one `/etc/group` file that contain all the users and groups on the network, and then copy these files to all the machines on the network.

  or

- Edit the `/etc/passwd` and `/etc/group` files on each machine to ensure that the following conditions are true:
  - Each user has the same user ID on all machines where that user has an account.
  - No two users anywhere on the network have the same user ID.
  - Each group has the same group ID on all machines where that group exists.
  - No two groups on the network have the same group ID.

When users request NFS access to remote files, their user IDs and group IDs are used to check file ownership and permissions, just as they are locally.

If a user has one user ID on an NFS client and a different user ID on an NFS server, the server will not grant the user access to his or her files on the server, because it thinks the files belong to someone else.

If a user on one machine has the same user ID as a user on another machine, one user may gain access to the other user's files.

For information on the `/etc/passwd` and `/etc/group` files, type `man 4 passwd` or `man 4 group` at the HP-UX prompt.

If you are using NIS, the `/etc/passwd` and `/etc/group` files are managed by a master server, and all other machines on the network request user and group information from the servers. With NIS, it is unnecessary to set user IDs and group IDs on each machine. For instructions on configuring NIS, see "Configuring and Administering NIS" on page 101.

To Ensure that No User is a Member of Too Many Groups

1. If you are not running NIS, issue the following command for each user on your system:

   `/usr/bin/grep -c` *`username`* `/etc/group`

   This command returns the number of occurrences of *username* in the `/etc/group` file.

   If you are using NIS to manage your `group` database, issue the following command for each user in your domain:

   `/usr/bin/ypcat -k group | /usr/bin/grep -c` *`username`*

   This command returns the number of occurrences of *username* in the NIS `group` database.

2. If any user is a member of more than 16 groups, remove the user from some of the groups. See “To Modify an NIS Map” on page 120 for instructions on modifying an NIS map.

   If you are running a version of HP-UX older than release 9.0, a user can be a member of only 8 groups, rather than 16.

If a user is a member of too many groups, NFS returns an RPC authentication error when the user attempts access to files or directories using NFS.

Configuring and Administering an NFS Server

An **NFS server** is a machine that "exports" its local directories (makes them available for client machines to mount using NFS) On the NFS client, these mounted files and directories look to users like part of the client's local file system. An NFS server can also be an NFS client. Following are the tasks involved in configuring and administering an NFS server. The first two tasks are the only ones required to get your server up and running.

- To Make Directories Available to NFS Clients (Export Directories)
- To Enable NFS Server Capability
- To Remove (Unexport) an Exported Directory
- To Enable PC NFS Server Capability
- To Disable NFS Server Capability

This section tells you how to perform these tasks, by editing files and issuing HP-UX commands. However, Hewlett-Packard recommends that you use SAM to configure and administer NFS. SAM (System Administration Manager) is Hewlett-Packard's windows-based user interface for performing system administration tasks. To run SAM, type `sam` at the HP-UX prompt. SAM has an extensive online help facility.

To Make Directories Available to NFS Clients (Export Directories)

1. Add a line to the `/etc/exports` file for each directory you want to make available to NFS clients, using a text editor like `vi`. If the `/etc/exports` file does not exist on your system, you will have to create it. Following is the syntax of a line in the `/etc/exports` file:

   directory `[-`*option*`[,`*option*`]]`

   Type `man 4 exports` at the HP-UX prompt for a complete list of the export options. After adding your exported directories to the `/etc/exports` file, you must enable NFS server capability before NFS clients can mount your exported directories. See "To Enable NFS Server Capability" on page 27.

2. If your system is already running as an NFS server, issue the following command to add the directory to your server's internal list of exported directories:

   `/usr/sbin/exportfs` *directory*

You can issue the `exportfs -i` command to add the directory to your server's internal list of exported directories, without adding the directory to the `/etc/exports` file. However, it will stop being exported when you reboot your system or restart NFS, unless you also add it to the `/etc/exports` file. (Issuing the `exportfs` command does not change the contents of the `/etc/exports` file.) Type `man 1M exportfs` for more information.

You cannot export a directory and its ancestor or descendant, if they are on the same disk or logical volume. For example, if you are exporting the root directory (`/`), you cannot also export `/opt`, unless `/` and `/opt` are on different disks or logical volumes. Likewise, if you are exporting `/opt/frame`, you cannot also export `/opt` unless `/opt/frame` and `/opt` are on different disks or logical volumes. However, if a directory and its ancestor or descendant *are* on different disks or logical volumes, and you want to export both of them, you *must* export them using two separate entries in `/etc/exports`. Use the `bdf`(1M) command to determine whether your file systems are on different disks or logical volumes. Each line in the `bdf` output is a separate disk or volume that requires its own entry in `/etc/exports` if you want to export it.

The `/etc/exports` file should be owned by root and have mode 644 (`-rw-r--r--`).

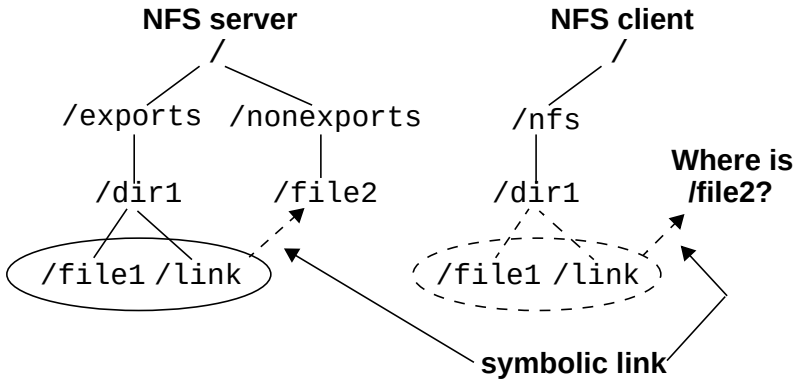
The export options that restrict access to an exported directory are applied in addition to the regular HP-UX permissions already in place on that directory. For example, if only the owner of a file has permission to write to it, nobody else can write to the file, even if it is exported to the world with read/write permission.

Access permissions may also be specified on the NFS client when a directory is mounted. If these permissions are different from the permissions for the exported directory on the NFS server, the more restrictive permissions are used.

It is not a good idea to export a directory if it contains a symbolic link that points outside the exported directory. Once the directory is mounted on an NFS client, the symbolic link will be resolved locally on the client, so the destination of the symbolic link must exist on the client as well as the server. If the destination of the symbolic link does not exist on the client, a `No such file or directory` message will be displayed whenever anyone attempts access to it.

Figure 2-1 illustrates the problem of symbolic links in NFS mounts, where the destination of the symbolic link exists on the NFS server but might not exist on the NFS client.

Figure 2-1 Symbolic Links in NFS Mounts



Examples from /etc/exports

The following example exports the /usr/bin directory to NFS clients cabbage, cauliflower, and broccoli. Users on client broccoli have read/write access to the /usr/bin directory. Users on cabbage and cauliflower have read-only access. In addition to the export options, the HP-UX permissions for the /usr/bin directory must be set to allow access to the world or to a group that includes the users on broccoli, cabbage and cauliflower.

```
/usr/bin -access=cabbage:cauliflower:broccoli,rw=broccoli
```

The following example allows all NFS clients read-only access to the directory /usr/share/man. The /usr/share/man directory must also allow read access to NFS users (for example, with -r--r--r-- permissions).

```
/usr/share/man -ro
```

The following example exports the /var/mail directory. It allows root access to clients sage, thyme, and basil. The root users on all other NFS clients are considered "unknown" to the NFS server, so they are given the access privileges of user nobody. Non-root users on all NFS clients are allowed read/write access to the /var/mail directory, if the HP-UX permissions on the /var/mail directory allow them read/write access.

```
/var/mail -root=sage:thyme:basil
```

The following example exports the private root directory of diskless client sage. It allows root access to the root user on client sage. All other users on client sage have read/write access, if they are allowed read/write access through the regular HP-UX permissions. Users on other NFS clients have read-only access, if they are allowed read access through the HP-UX permissions.

```
/export/private_roots/sage -rw=sage,root=sage
```

In the following example, any user without a valid user ID who attempts access to client basil's private root directory will receive an RPC authentication error, because anonymous access is denied with the anon=65535 option. The root user on client basil is allowed root access to the directory, but the root users on all other machines are treated as "unknown" and denied access. The non-root users on all NFS clients are allowed read/write access, if the HP-UX permissions on that directory allow them read/write access.

```
/export/private_roots/basil -root=basil,anon=65535
```

The following example exports the `/export/newsletter` directory to all NFS clients. Root users will be given the effective user ID of 200. Other anonymous users will keep their own user IDs (even though they do not exist in the NFS server's `passwd` database), but they will be given the access permissions associated with user ID 200. If a root user is allowed to create a file in this directory, the `ls` command will show that it is owned by user ID 200. If an anonymous user with a non-zero user ID (for example, 840) is allowed to create a file in this directory, the `ls` command will show that it is owned by user ID 840.

```
/export/newsletter -anon=200
```

The following example exports the `/opt/frame` directory to all NFS clients. Non-root users have read/write access (if the regular HP-UX permissions allow it), and root users are given the access privileges of user `nobody`. NFS writes are done asynchronously; that is, when an NFS client writes data to a mounted directory, the server returns a response before writing the data to disk. This allows the client to continue processing without waiting for the write request to complete.

```
/opt/frame -async
```

To Enable NFS Server Capability

1. In the `/etc/rc.config.d/nfsconf` file, make sure the `NFS_SERVER` and `START_MOUNTD` variables are set to 1, as follows:

   ```
   NFS_SERVER=1
   START_MOUNTD=1
   ```

2. Issue the following command to run the NFS startup script:

   ```
   /sbin/init.d/nfs.server start
   ```

The NFS startup script uses the variables in `/etc/rc.config.d/nfsconf` to determine which processes to start.

The `START_MOUNTD` variable causes the NFS startup script to start `rpc.mountd`, the mount daemon.

CAUTION If `rpc.mountd` is configured in `/etc/inetd.conf` on your system, set the `START_MOUNTD` flag to 0. Mounts will fail if `rpc.mountd` is enabled through both `/etc/inetd.conf` and `/etc/rc.config.d/nfsconf`.

For more information, see the following man pages: `mountd`(1M) and `inetd.conf`(4).

To Remove (Unexport) an Exported Directory

1. On the NFS server, issue the following command for a list of all the NFS clients that have mounted the directory you want to unexport:

   ```
   /usr/sbin/showmount -a
   ```

NOTE

The output of the `showmount` command is not always complete. If an NFS client mounts a remote directory twice and unmounts it only once, the remote directory is still mounted on the client, but the `showmount` command does not list that client. Also, clients configured to automount a directory will not be listed by the `showmount` command if the directory is not currently mounted.

2. On every NFS client that has the directory mounted, issue the following command for a list of the process IDs and user names of everyone using the mounted directory:

   `/usr/sbin/fuser -u` *`servername`*`:/`*`directory`*

3. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can use the following command to kill all processes using the directory:

   `/usr/sbin/fuser -ck` *`local_mount_point`*

4. On every NFS client that has the directory mounted, issue the following command to unmount the directory:

   `/usr/sbin/umount` *`local_mount_point`*

   or

   `/usr/sbin/umount` *`servername`*`:/`*`directory`*

5. On every NFS client that had the directory mounted, use a text editor to comment out or remove the line in the `/etc/fstab` file that lists the directory you want to unexport. This prevents clients from attempting to mount the directory when they reboot.

6. On every client that has the directory configured to be automounted, edit the `/etc/auto_*` files to comment out or remove the directory from the automounter maps. Clients that automount the directory may not be listed by the `showmount` command.

   If you are using NIS to manage your automounter maps, edit the `/etc/auto_*` files on the NIS master server, and then issue the following commands to regenerate the maps and push them to the slave servers:

```
cd /var/yp
/usr/ccs/bin/make auto_mapname auto_mapname ...
```

7. If you modified the automounter master map, or if you added or deleted an entry in an automounter direct map, issue the following command, on all clients that use the map, to force AutFS to reread its maps:

   ```
   /usr/sbin/automount
   ```

8. On the NFS server, use a text editor to remove the line in the `/etc/exports` file that lists the directory you want to unexport.

9. On the NFS server, issue the following command to unexport the directory:

   ```
   /usr/sbin/exportfs -u directory
   ```

If you unexport a directory that an NFS client currently has mounted, the next time someone on that client requests access to the directory, NFS will return an `NFS stale file handle` error message. The client may be able to unmount the directory, but if that does not work, the client must reboot to recover.

For more information, see the following man pages: `showmount`(1M), `fuser`(1M), `umount`(1M), and `exportfs`(1M), `automount`(1M), `make`(1), and `ypmake`(1M).

To Enable PC NFS Server Capability

1. If necessary, create a file called `/etc/pcnfsd.conf` and add PC NFS configuration information to it. The `/etc/pcnfsd.conf` file is not required in order to run `pcnfsd`. For more information on the `/etc/pcnfsd.conf` file, type `man 1M pcnfsd` at the HP-UX prompt.
2. In the `/etc/rc.config.d/nfsconf` file, use a text editor to set the `PCNFS_SERVER` flag to 1, as follows:

   ```
   PCNFS_SERVER=1
   ```
3. Issue the following command to run the NFS startup script:

   ```
   /sbin/init.d/nfs.server start
   ```

The `PCNFS_SERVER` flag causes the NFS startup script to start the PC NFS server daemon, `pcnfsd`. As a PC NFS server, your system can export its directories and files to PC NFS clients.

Following are some reasons why you might want to create an `/etc/pcnfsd.conf` file:

- If your PC NFS client software is assigning user IDs smaller than 101 or greater than 60002, set the `uidrange` in the `/etc/pcnfsd.conf` file to allow access to a different range of user IDs, as in the following example:

  ```
  uidrange 80-60005
  ```
- If you want to give PC users a different set of default print options, the `/etc/pcnfsd.conf` file should contain a line similar to the following, which defines `raw` as a default print option for PC users submitting jobs to the printer `lj3_2`:

  ```
  printer lj3_2 lj3_2 lp -dlj3_2 -oraw
  ```

The `/etc/pcnfsd.conf` file is read when the `pcnfsd` daemon starts up. If you make any changes to `/etc/pcnfsd.conf` while `pcnfsd` is running, you must restart `pcnfsd` before your changes will take effect.

A PC must have NFS client software installed in order to use your system as a PC NFS server.

For more information on `pcnfsd`, type `man 1M pcnfsd` at the HP-UX prompt.

To Disable NFS Server Capability

1. On the NFS server, issue the following command for a list of all the NFS clients that have directories mounted from the NFS server you are planning to disable:

   ```
   /usr/sbin/showmount -a
   ```

NOTE

The output of the `showmount` command is not always complete. If an NFS client mounts a remote directory twice and unmounts it only once, the remote directory is still mounted on the client, but the `showmount` command does not list that client. Also, clients that are configured to automount a directory will not be listed by the `showmount` command if the directory is not currently mounted.

2. On every NFS client listed by the `showmount` command, issue the following command for each directory that is mounted from your NFS server:

   `/usr/sbin/fuser -u` *`servername:/directory`*

   This command lists the process IDs and user names of everyone using the mounted directory.

3. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can use the following command to kill all processes using the directory:

   `/usr/sbin/fuser -ck` *`local_mount_point`*

4. On every client that has directories mounted from your server, issue the following command:

   `/usr/sbin/umount -h` *`servername`*

5. If your server will be down for a long time, edit the `/etc/fstab` file on each client to comment out or remove any NFS mounts from the server you are planning to disable. This prevents the clients from attempting to mount directories from your server when the clients are rebooted.

6. If your server will be down for a long time, edit the `/etc/auto_*` files on each client to comment out or remove any automounts from the server you are planning to disable. Clients that automount the server's directories might not be listed by the `showmount` command.

If you are using NIS to manage your automounter maps, edit the `/etc/auto_*` files on the NIS master server, and then issue the following commands to regenerate the maps and push them to the slave servers:

```
cd /var/yp
/usr/ccs/bin/make auto_mapname auto_mapname ...
```

7. If you modified the automounter master map, or if you added or deleted any entries in an automounter direct map, issue the following command, on all clients that use the map, to force AutoFS to reread its maps:

   ```
   /usr/sbin/automount
   ```

8. Issue the following command on the server to unexport all exported directories:

   ```
   /usr/sbin/exportfs -au
   ```

9. On the NFS server, edit the `/etc/rc.config.d/nfsconf` file to set the `NFS_SERVER` variable to 0. This prevents the NFS server daemons from starting up when your system reboots. If your server will be down only a short time, this step is unnecessary.

   ```
   NFS_SERVER=0
   ```

10. Edit the `/etc/inetd.conf` file to comment out the line that contains `rpc.mountd` (if it exists) and the lines for the other RPC services.

11. Issue the following command to disable NFS server capability:

    ```
    /sbin/init.d/nfs.server stop
    ```

If your NFS server will be down for only a very short period of time, this procedure is not necessary. If the server is down for only a few minutes, and directories are hard-mounted on the clients, clients attempting access to the server will simply hang until it comes back up. Then, they will resume access to it as if nothing had happened.

However, if the server will be down for a long time, NFS clients attempting access to it will have to interrupt their attempts, usually with `[CTRL]-C`. If directories are mounted with the `nointr` option, clients must reboot their systems in order to stop trying to access a down server.

See the following man pages for more information: `showmount`(1M), `fuser`(1M), `exportfs`(1M), `automount`(1M), and `mountd`(1M).

Configuring and Administering an NFS Client

An **NFS client** is a machine that "mounts" remote directories using NFS. These mounted remote directories appear to users as if they are part of the NFS client's local file system. An NFS client can also be an NFS server. Following are the tasks involved in configuring and administering an NFS client. Only the first four tasks are required in order to get your client up and running.

- To Decide Between Standard-Mounted and Automounted Directories
- To Mount a Remote Directory Using a Standard NFS Mount
- To Enable NFS Client Capability
- To Verify Your NFS Client Configuration
- To Change the Default Mount Options
- To Ensure Data Integrity Between the Client and Server
- To Remove (Unmount) a Mounted Directory
- To Disable NFS Client Capability

This section tells you how to perform these tasks, by editing files and issuing HP-UX commands. However, Hewlett-Packard recommends that you use SAM to configure and administer NFS. SAM (System Administration Manager) is Hewlett-Packard's windows-based user interface for performing system administration tasks. To run SAM, type `sam` at the HP-UX prompt. SAM has an extensive online help facility.

To Decide Between Standard-Mounted and Automounted Directories

- Before you mount any remote directories on your local system, decide whether you want each directory to be standard-mounted or automounted. Table 2-1 lists the advantages and disadvantages of each type of mount. For instructions on automounting remote directories, see "Configuring and Administering AutoFS" on page 51.

  Standard-mounted directories stay mounted until you explicitly unmount them. Automounted directories stay mounted until they are left idle for five minutes. The five minute default can be changed by adding the `-t` *duration* option to the `AUTOMOUNT_OPTIONS` variable in the `/etc/rc.config.d/nfsconf` file.

Table 2-1 Standard-Mounted vs. Automounted Directories

Standard-Mounted Directory	Automounted Directory
Advantage: Configuration is simpler than for automounted directories. Only one file (`/etc/fstab`) is used to configure standard mounts.	*Disadvantage:* Configuration can be more complicated than for standard mounts. Multiple files are usually required to configure AutoFS.
Advantage: The directory stays mounted, so you never have to wait for it to be mounted after you issue a read or write request.	*Disadvantage:* If the automounted directory has timed out and been unmounted, and you attempt to read it or write to it, you may have to wait a few seconds for it to be mounted again.
Disadvantage: If a directory is configured to be standard-mounted when your system boots, and the NFS server for the directory is not booted yet, your system will hang until the NFS server becomes available. If your system and the server are configured to mount directories from each other at boot time, standard mounts can cause both systems to hang indefinitely.	*Advantage:* An automounted directory is not mounted until a user or process requests access to it, so both your system and the NFS server will have time to boot before any attempt is made to mount the directory.
Disadvantage: The configuration file for standard mounts (`/etc/fstab`) must be maintained separately on each NFS client.	*Advantage:* AutoFS configuration files (maps) may be managed centrally through NIS.

Standard-Mounted Directory	Automounted Directory
Disadvantage: Only one NFS server may be configured for each standard-mounted directory.	*Advantage:* Multiple servers may be configured for a single automounted directory, for reliability and load balancing. All servers are polled simultaneously, and the directory is mounted from the first server to respond.
Disadvantage: If you have to configure many similar standard mounts, you must configure each of them individually, because you cannot use wildcard characters or environment variables when you configure standard NFS mounts.	*Advantage:* AutoFS allows you to use wildcard characters and environment variables in configuration files (maps) as shortcuts when you are configuring many similar automounts.
Disadvantage: Standard NFS mounts provide no shortcut for configuring all available remote directories; each directory must be configured explicitly. If the NFS servers change which directories they are exporting, you must change your local NFS client configuration.	*Advantage:* AutoFS allows you to configure a special "built-in" map (the `-hosts` map), which causes all the exported directories from any NFS server on the network to be automounted on your system whenever anyone requests access to a directory on that server. The servers can change which directories they export, and your configuration remains valid.

To Mount a Remote Directory Using a Standard NFS Mount

1. In the `/etc/fstab` file, use a text editor to add a line for each remote directory you want mounted on your system. If the `/etc/fstab` file does not exist, you will have to create it. A line in the `/etc/fstab` file has the following syntax:

   server`:`*remote_directory* *local_directory* `nfs defaults 0 0`

   or

   server`:`*remote_directory* *local_directory* `nfs` *option*`[,`*option*`...] 0 0`

   For descriptions of the mount options, see "To Change the Default Mount Options" on page 40.

2. If your system is already running as an NFS client, issue the following command to mount each remote directory you have added to the `/etc/fstab` file:

   `/usr/sbin/mount` *local_directory*

   Or, issue the following command to mount all the directories listed in the `/etc/fstab` file:

   ```
   /usr/sbin/mount -a
   ```

The remote directories listed in the `/etc/fstab` file will be mounted automatically when you enable NFS client capability or reboot your system. See "To Enable NFS Client Capability" on page 39.

The local directory you configure as a mount point must exist and should be empty. If the local mount point contains files or directories, they will be hidden and inaccessible while the remote directory is mounted over them.

Before you can mount a remote directory on your system, the remote system where the directory is located must be configured as an NFS server and must export the directory.

To mount a directory temporarily, issue the `mount` command, but do not add the mount to the `/etc/fstab` file. It will stay mounted until you reboot your system or until you unmount it with the `umount` command.

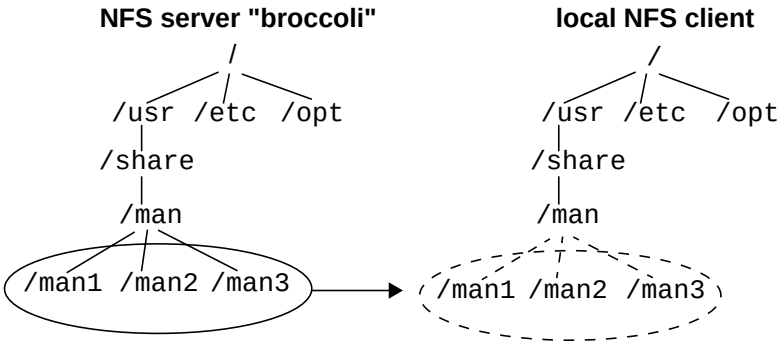
For more information, type `man 4 fstab` or `man 1M mount` at the HP-UX prompt.

Example NFS Mount of man pages

```
broccoli:/usr/share/man /usr/share/man NFS ro 0 0
```

This example mounts the directory `/usr/share/man` from the NFS server `broccoli`. The local mount point is also `/usr/share/man`. The directory is mounted read-only. Figure 2-2 illustrates this example:

Figure 2-2 NFS Mount of man pages

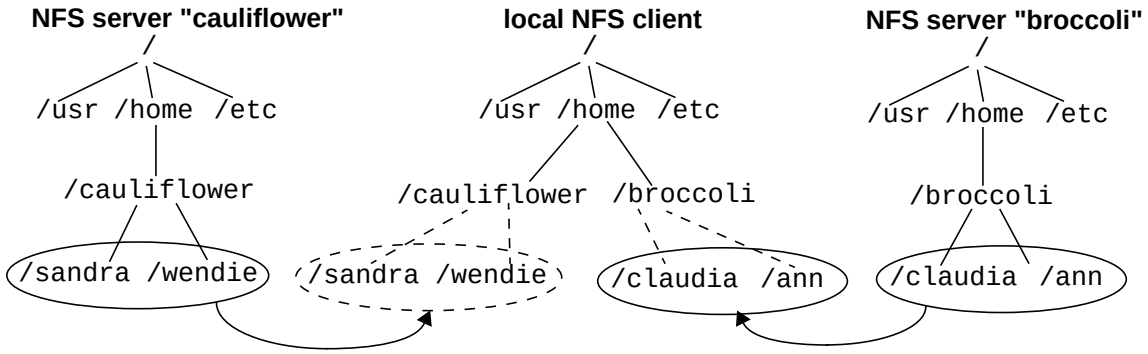


Example NFS Mount of Home Directories

```
broccoli:/home/broccoli /home/broccoli nosuid 0 0
cauliflower:/home/cauliflower /home/cauliflower nosuid 0 0
```

This example mounts the home directories from NFS servers `broccoli` and `cauliflower` on the local NFS client. The `nosuid` option prevents programs with setuid permission from executing on the local client. Figure 2-3 illustrates this example:

Figure 2-3 NFS Mount of Home Directories



To Enable NFS Client Capability

1. In the `/etc/rc.config.d/nfsconf` file, make sure the `NFS_CLIENT` variable is set to 1, as follows:

   ```
   NFS_CLIENT=1
   ```

2. Run the NFS startup script by issuing the following command:

   ```
   /sbin/init.d/nfs.client start
   ```

Setting the `NFS_CLIENT` variable to 1 causes the NFS startup script to be run whenever you reboot your system.

The NFS startup script starts the necessary NFS client daemons and mounts the remote directories configured in the `/etc/fstab` file.

To Verify Your NFS Client Configuration

- After you have configured the directories you want to mount and enabled NFS client capability, issue the `ls` command in the local directories you have configured as NFS mount points. If your NFS client is working correctly, the `ls` command will list the contents of mounted directories. If the local directories are empty, or if you get error messages, see “Troubleshooting NFS Services” on page 173.

To Change the Default Mount Options

1. Include the NFS mount options in your `/etc/fstab` file or automounter map as needed. Table 2-2 and Table 2-3 list the NFS mount options.
2. If you changed the mount options in the automounter master map, you must run the `automount`(1M) command, on each client that uses the map, before your changes will take effect.

   If you changed the mount options for a directory that is currently mounted, you must unmount and remount it before your changes will take effect. Issue the following commands:

   `/usr/sbin/umount` *local_directory*
   `/usr/sbin/mount` *local_directory*

Table 2-2 NFS Mount Options

`rw` (read/write) or `ro` (read-only) (default: `rw`)	Use `rw` for data that users need to modify. In order for you to mount a directory read/write, the NFS server must export it read/write. Use `ro` for data you do not want users to change. A directory that is automounted from several servers should be read-only, to keep versions identical on all servers.
`suid` or `nosuid` (default: `suid`)	Specify `suid` if you want to allow mounted programs that have setuid permission to run with the permissions of their owners, regardless of who starts them. If a program with setuid permission is owned by root, it will run with root permissions, regardless of who starts it. Specify `nosuid` to protect your system against setuid programs that may run as root and damage your system.

`hard`
or
`soft`
 	
(default: `hard`)
	Specify `hard` if users will be writing to the mounted directory or running programs located in it. When NFS tries to access a hard-mounted directory, it keeps trying until it succeeds or someone interrupts its attempts. If the server goes down, any processes using the mounted directory hang until the server comes back up and then continue processing without errors. Interruptible hard mounts may be interrupted with `CTRL-C` or `kill` (see the `intr` option, later).
	
	Specify `soft` if the server is unreliable and you want to prevent systems from hanging when the server is down. When NFS tries to access a soft-mounted directory, it gives up and returns an error message after trying `retrans` times (see the `retrans` option, later). Any processes using the mounted directory will return errors if the server goes down.

`intr`
or
`nointr`
 	
(default: `intr`)
	Specify `intr` if users are not likely to damage critical data by manually interrupting an NFS request. If a hard mount is interruptible, a user may press `[CTRL]-C` or issue the `kill` command to interrupt an NFS mount that is hanging indefinitely because a server is down.
	
	Specify `nointr` if users might damage critical data by manually interrupting an NFS request, and you would rather have the system hang while the server is down than risk losing data between the client and the server.

`fg` (foreground) or `bg` (background) (default: `fg`)	Specify `fg` for directories that are necessary for the client machine to boot or operate correctly. If a foreground mount fails, it is retried again in the foreground until it succeeds or is interrupted. All automounted directories are mounted in the foreground; you cannot specify the `bg` option with automounted directories. Specify `bg` for mounting directories that are not necessary for the client to boot or operate correctly. Background mounts that fail are retried in the background, allowing the mount process to consider the mount complete and go on to the next one. If you have two machines configured to mount directories from each other, configure the mounts on one of the machines as background mounts. That way, if both systems try to boot at once, they will not become deadlocked, each waiting to mount directories from the other. The `bg` option cannot be used with automounted directories.

`devs` or `nodevs` (default: `devs`)	Specify `devs` if you are mounting device files from a server whose device files will work correctly on the client. The `devs` option allows you to use NFS-mounted device files to read and write to devices from the NFS client. It is useful for maintaining a standard, centralized set of device files, if all your systems are configured similarly. Specify `nodevs` if device files mounted from a server will not work correctly for reading and writing to devices on the NFS client. The `nodevs` option generates an error if a process on the NFS client tries to read or write to an NFS-mounted device file.
`timeo=`*n* (default=7)	The timeout, in tenths of a second, for NFS requests (read and write requests to mounted directories). If an NFS request times out, this timeout value is doubled, and the request is retransmitted. After the NFS request has been retransmitted the number of times specified by the `retrans` option (see below), a soft mount returns an error, and a hard mount retries the request. The maximum `timeo` value is 30 (3 seconds). Try doubling the `timeo` value if you see several `server not responding` messages within a few minutes. This can happen because you are mounting directories across a gateway, because your server is slow, or because your network is busy with heavy traffic.
`retrans=`*n* (default=4)	The number of times an NFS request (a read or write request to a mounted directory) is retransmitted after it times out. If the request does not succeed after *n* retransmissions, a soft mount returns an error, and a hard mount retries the request. Increase the `retrans` value for a directory that is soft-mounted from a server that has frequent, short periods of down time. This gives the server sufficient time to recover, so the soft mount does not return an error.
`retry=`*n* (default=1)	The number of times the NFS client attempts to mount a directory after the first attempt fails. If you specify `intr`, you can interrupt the mount before *n* retries. However, if you specify `nointr`, you must wait until *n* retries have been made, until the mount succeeds, or until you reboot the system. If mounts are failing because your server is very busy, increasing the `retry` value may fix the problem.

`rsize=`*n* (default=8192)	The number of bytes the NFS client requests from the NFS server in a single read request. If packets are being dropped between the client and the server, decrease `rsize` to 4096 or 2048. To find out whether packets are being dropped, issue the `nfsstat -rc` command at the HP-UX prompt. If the `timeout` and `retrans` values returned by this command are high, but the `badxid` number is close to zero, then packets are being dropped somewhere in the network.
`wsize=`*n* (default=8192)	The number of bytes the NFS client sends to the NFS server in a single write request. If packets are being dropped between the client and the server, decrease `wsize` to 4096 or 2048. To find out whether packets are being dropped, issue the `nfsstat -rc` command at the HP-UX prompt. If the `timeout` and `retrans` values returned by this command are high, but the `badxid` number is close to zero, then packets are being dropped somewhere in the network.
`vers=`*n* (default=3)	The version of the NFS protocol to use. By default, the local NFS client will attempt to mount the file system using NFS version 3. If the NFS server does not support version 3, the file system will be mounted using version 2. If you know that the NFS server does not support version 3, specify `vers=2`, and you will save time during the mount, because the client will not attempt to use version 2 before using version 2.
`O` (Overlay mount) default: not specified	Allows the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If you attempt to mount a file system over an existing mount point without the `-O` option, the mount will fail with the error `device busy`. *Caution*: Using the `-O` mount option can put your system in a confusing state. The `-O` option allows you to hide local data under an NFS mount point without receiving any warning. Local data hidden beneath an NFS mount point will not be backed up during regular system backups. On HP-UX, the `-O` option is valid only for NFS-mounted file systems. For this reason, if you specify the `-O` option, you must also specify the `-F nfs` option to the `mount` command or the `nfs` file system type in the `/etc/fstab` file.

`remount` default: not specified	If the file system is mounted read-only, this option remounts it read/write. This allows you to change the access permissions from read-only to read/write without forcing everyone to leave the mounted directory or killing all processes using it.
`grpid` default: not specified	Forces a newly created file in the mounted file system to inherit the group ID of the parent directory. By default, a newly created file inherits the effective group ID of the calling process, unless the GID bit is set on the parent directory. If the GID bit is set, the new file inherits the group ID of the parent directory.

Several NFS mount options allow you to change the length of time file and directory attributes remain cached on the NFS client. By default, an NFS client caches certain attributes of files and directories, like their ownership, size, and modification time. If a user on an NFS client is making a series of changes to a file, the changes to the file's attributes are cached and modified locally on the client, and finally, the resulting attributes are sent to the server.

Table 2-3 NFS Caching Options

`noac` (default: not specified)	If specified, this option prevents the NFS client from caching attributes for the mounted directory. Specify `noac` for a directory that will be used frequently by many NFS clients. The `noac` option ensures that the file and directory attributes on the server are up to date, because no changes are cached on the clients. However, if many NFS clients using the same NFS server all disable attribute caching, the server may become overloaded with attribute requests and updates. You can also use the `actimeo` option to set all the caching timeouts to a small number of seconds, like 1 or 3. If you specify `noac`, do not specify the other caching options.
`nocto` (default: not specified)	If specified, this option suppresses fresh attributes when opening a file. Specify `nocto` for a file or directory that never changes, to decrease the load on your network.
`acdirmax=`*n* (default=60)	The maximum number of seconds a directory's attributes are cached on the NFS client. When this timeout period expires, the client flushes its attribute cache, and if the attributes have changed, the client sends them to the NFS server. For a directory that rarely changes or that is owned and modified by only one user, like a user's home directory, you can decrease the load on your network by setting `acdirmax=120` or higher.
`acdirmin=`*n* (default=30)	The minimum number of seconds a directory's attributes are cached on the NFS client. If the directory is modified before this timeout expires, the timeout period is extended by `acdirmin` seconds. For a directory that rarely changes or that is owned and modified by only one user, like a user's home directory, you can decrease the load on your network by setting `acdirmin=60` or higher.

`acregmax=`*n* (default=60)	The maximum number of seconds a file's attributes are cached on the NFS client. When this timeout period expires, the client flushes its attribute cache, and if the attributes have changed, the client sends them to the NFS server. For a file that rarely changes or that is owned and modified by only one user, like a file in a user's home directory, you can decrease the load on your network by setting `acregmax=120` or higher.
`acregmin=`*n* (default=3)	The minimum number of seconds a file's attributes are cached on the NFS client. If the file is modified before this timeout expires, the timeout period is extended by `acregmin` seconds. For a file that rarely changes or that is owned and modified by only one user, like a file in a user's home directory, you can decrease the load on your network by setting `acdirmin=30` or higher.
`actimeo=`*n* (no default)	Setting `actimeo` to *n* seconds is equivalent to setting `acdirmax`, `acdirmin`, `acregmax`, and `acregmin` to *n* seconds. Set `actimeo=1` or `actimeo=3` for a directory that is used and modified frequently by many NFS clients. This ensures that the file and directory attributes are kept reasonably up to date, even if they are changed frequently from various client locations. Set `actimeo=120` or higher for a directory that rarely or never changes. If you set the `actimeo` value, do not set the `acdirmax`, `acdirmin`, `acregmax`, or `acregmin` values.

To Ensure Data Integrity Between the Client and Server

- Make sure the directory is exported from the server with the `noasync` option (the default). If the directory is exported with the `async` option, the NFS server will acknowledge NFS writes before writing data to disk. Changing an exported directory from `async` to `noasync` degrades write performance for that directory.
- If users or applications will be writing to the NFS-mounted directory, make sure it is mounted with the `hard` option (the default), rather than the `soft` option.
- If you have a small number of NFS applications that require absolute data integrity, add the `O_SYNC` flag to the `open()` calls in your applications. When you open a file with the `O_SYNC` flag, a `write()` call will not return until the write request has been sent to the NFS server and acknowledged. The `O_SYNC` flag degrades write performance for applications that use it.
- If you have a large number of NFS applications requiring absolute data integrity, or if your entire installation needs a high degree of data integrity, set the `NUM_NFSIOD` variable to 0 in the `/etc/rc.config.d/nfsconf` file on each client, as follows,

  ```
  NUM_NFSIOD=0
  ```

  and issue the following commands to kill all the `biod` daemons (*PID* is a process ID returned by the `ps` command):

  ```
  /usr/bin/ps -ef | /usr/bin/grep biod
  /usr/bin/kill PID PID ...
  ```

  The `biod` daemons improve write performance by handling NFS write requests from users and applications. After a write request is passed to a `biod` daemon, control is returned to the user or application. Running a client without `biod` daemons degrades write performance for all users and applications on that client.
- If multiple NFS users will be writing to the same file, add the `lockf()` call to your applications to lock the file so that only one user may modify it at a time.

  If multiple users on different NFS clients will be writing to the file, you must also turn off attribute caching on those clients by mounting the file with the `noac` option.

For more information, see the following man pages: `mount`(1M), `open`(2), `write`(2), `lockf`(2), and `biod`(1M).

To Remove (Unmount) a Mounted Directory

1. On the NFS client, issue the following command to determine whether the directory you want to unmount is currently in use:

   `/usr/sbin/fuser -cu` *local_mount_point*

   This command lists the process IDs and user names of everyone using the mounted directory.

2. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can use the following command to kill all processes using the mounted directory:

   `/usr/sbin/fuser -ck` *local_mount_point*

3. If you want to remove the mounted directory permanently, use an editor to remove the appropriate line in the `/etc/fstab` file.

   If you want to remove the mounted directory temporarily, leave the line in `/etc/fstab`, and the directory will be mounted again when you reboot your system or run the NFS startup script.

4. Issue the following command at the HP-UX prompt:

   `/usr/sbin/umount` *local_mount_point*

   If any user or process is using the remote directory, NFS cannot unmount it and will issue an error message.

For more information, type `man 1M mount` or `man 1M fuser` at the HP-UX prompt.

To Disable NFS Client Capability

1. On the NFS client, issue the `mount`(1M) command with no options, to get a list of all the mounted file systems on the client:

   ```
   /usr/sbin/mount
   ```

2. For every NFS-mounted directory listed by the `mount` command, issue the following command to determine whether the directory is currently in use:

   ```
   /usr/sbin/fuser -cu local_mount_point
   ```

   This command lists the process IDs and user names of everyone using the mounted directory.

3. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can use the following command to kill all processes using the mounted directory:

   ```
   /usr/sbin/fuser -ck local_mount_point
   ```

4. Issue the following command on the client to unmount all NFS-mounted directories:

   ```
   /usr/sbin/umount -at nfs
   ```

5. Edit the `/etc/rc.config.d/nfsconf` file on the client to set the `NFS_CLIENT` and `AUTOMOUNT` variables to 0. This prevents the client processes from starting up again when you reboot the client.

   ```
   NFS_CLIENT=0
   AUTOMOUNT=0
   ```

6. Issue the following command to disable NFS client capability:

   ```
   /sbin/init.d/nfs.client stop
   ```

For more information, type `man 1M mount` or `man 1M fuser` at the HP-UX prompt.

Configuring and Administering AutoFS

This section tells you how to configure AutoFS. AutoFS mounts directories automatically when users or processes request access to them, and it unmounts them automatically after they have been idle for a period of time (five minutes, by default). Following are the tasks involved in configuring AutoFS. Tasks 3 and 16 alone will get AutoFS up and running on your system.

Before configuring AutoFS, see "To Decide Between Standard-Mounted and Automounted Directories" on page 34.

1. To Migrate from the Automounter to AutoFS
2. To Understand How AutoFS Works
3. To Automount All Exported Directories from Any Host Using the -hosts Map
4. To Decide Between Direct and Indirect NFS Automounts
5. To Mount a Remote Directory Using a Direct Automounter Map
6. To Mount a Remote Directory Using an Indirect Automounter Map
7. To Configure Multiple (Replicated) Servers for an Automounted Directory
8. To Use Environment Variables as Shortcuts in Automounter Maps
9. To Use Wildcard Characters as Shortcuts in Automounter Maps
10. To Automount Users' Home Directories
11. To Automount Multiple Directories Simultaneously (Hierarchical Mounts)
12. To Automount a Directory Using CacheFS
13. To Include an Automounter Map in Another Automounter Map
14. To Create a Hierarchy of Automounter Maps
15. To Turn Off an Automounter Map with the -null Map
16. To Enable AutoFS
17. To Disable AutoFS
18. To Verify Your AutoFS Configuration

19. To Modify or Remove (Unmount) an Automounted Directory

This section tells you how to perform these tasks, by editing files and issuing HP-UX commands. However, Hewlett-Packard recommends that you use SAM to configure and administer AutoFS. SAM (System Administration Manager) is Hewlett-Packard's windows-based user interface for performing system administration tasks. To run SAM, type `sam` at the HP-UX prompt. SAM has an extensive online help facility.

NOTE SAM does not support specifying maps or directories on the `automount` command line. SAM finds AutoFS maps only if they are listed in the master map. SAM recognizes automounted directories only if they are listed in an AutoFS map.

To Migrate from the Automounter to AutoFS

The 10.20 ACE and HWE replace the old automounter with AutoFS, which has the following advantages over the old automounter:

- AutoFS can be used to mount any type of file system, including NFS Protocol Version 3. (The old automounter can be used only for NFS PV2.)
- With AutoFS the configured mount points are the actual mount points. (The old automounter mounted directories under `/tmp_mnt` and creates symbolic links from the configured mount points to the actual ones under `/tmp_mnt`.)
- You do not have to stop AutoFS to change your automounter maps. The AutoFS daemon, `automountd`, runs continuously. When you make a change to an automounter map, you run the `automount` command, which reads the maps and then exits. (The old automounter had to be killed and restarted whenever you made a change to an automounter map.)

If you were using the automounter before you installed the 10.20 ACE or HWE, you must perform the following tasks to migrate your automounter configuration to AutoFS:

1. Move the `/etc/rc.config.d/nfsconf` file to `/etc/rc.config.d.nfsconf.old`.
2. Copy the `/usr/newconfig/etc/rc.config.d/nfsconf` file to `/etc/rc.config.d/nfsconf`.
3. Copy any options you had specified in the `AUTO_OPTIONS` variable to either the `AUTOMOUNT_OPTIONS` or the `AUTOMOUNTD_OPTIONS` variable. Remove obsolete options.

   The old `automount` daemon is replaced by the `automount`(1M) command and the `automountd`(1M) daemon. Each has its own set of options. Table 2-4 lists the options to the old `automount` command and the equivalent AutoFS command options. It also indicates which `automount` options are obsolete with AutoFS.

Table 2-4 Old Automount Command-Line Options Used By AutoFS

Old automount Option	Equivalent AutoFS Command Option	Purpose
`-D` *variable=value*	`automountd -D` *variable=value*	Assign *value* to environment *variable*.
`-f` *master_file*	`automount -f` *master_file*	Use *master_file* as local master map.
`-M` *mount_directory*	Obsolete with AutoFS.	Automount directories under *mount_directory* instead of */tmp_mnt*.
`-m`	Obsolete with AutoFS.	Ignore NIS `auto.master` map.
`-n`	Obsolete with AutoFS.	Allow automounts only of previously mounted target file systems.
`-T`	`automountd -T`	Enable `automount` tracing.
`-tl` *duration*	`automount -t` *duration*	Specify time before unmounting idle directories.
`-tm` *interval*	Obsolete with AutoFS.	Specify interval between mount attempts.
`-tw` *interval*	Obsolete with AutoFS.	Specify interval between unmount attempts.
-v	`automount -v` `automountd -v`	Verbose mode.

4. Modify any scripts you have that kill and restart `automount`. The new AutoFS daemon, `automountd`, rarely needs to be restarted. If you need to make changes to your automounter maps, just run the `automount` program after modifying the maps. It is not a daemon, like the old `automount` process; it is a program that runs once to read the maps and then terminates.

For more information, see the `automount`(1M) or `automountd`(1M) man page.

To Understand How AutoFS Works

AutoFS consists of the following components:

1. The `automount` command, for reading automounter maps into memory.
2. The AutoFS file system.
3. The `automountd` daemon, which automounts file systems when they are requested by users.

The `automount` command is invoked at system startup. It reads the automounter master map to create the initial set of AutoFS mount points in the internal mount table, `/etc/mnttab`. The automounted file systems are not automatically mounted at startup. They are points under which file systems will be mounted later, when users request access to them.

When AutoFS receives a request to mount a file system that is not currently mounted, it calls the `automountd` daemon, which actually mounts the requested file system. Once the file system is mounted, further access does not require any action from the `automountd` daemon. Unlike the old automounter, AutoFS mounts file systems at the configured mount points. It does not maintain its own directory of mount points with symbolic links into it the way the old automounter does.

The `automountd` daemon is completely independent from the `automount` command. Because of this separation, it is possible to add, delete, or change automounter map information without having to stop and restart the `automountd` daemon.

After system startup, when the AutoFS mount points are set up, you can modify the set of mount points by modifying the automounter maps and running the `automount` command to read them and modify the mount table accordingly. You do not have to stop and restart AutoFS.

If an automounted file system has been idle for 5 minutes, AutoFS unmounts it.

For more information on AutoFS, type `man 1M automount` or `man 1M automountd` at the HP-UX prompt.

To Automount All Exported Directories from Any Host Using the -hosts Map

1. If you are using local files for your automounter maps, use an editor to add the following line to the automounter master map file, /etc/auto_master:

   ```
   /net      -hosts      nosuid
   ```

   If you are using NIS to manage your automounter maps, add the line to the master map file on the NIS master server, and then issue the following commands to rebuild the map and push it out to slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make auto_master
   ```

2. On each host that will use the map you have just modified, issue the following command to force AutoFS to read the modified map:

   ```
   /usr/sbin/automount
   ```

The local mount point (/net) should not exist.

You must enable AutoFS before any directories can be automounted. See "To Enable AutoFS" on page 80.

The -hosts map is a "built-in" automounter map; you do not have to create it. The -hosts map causes AutoFS to mount all the exported directories from any NFS server on the network whenever a user or process requests access to one of the exported directories from that server.

CAUTION Because the -hosts map allows NFS access to any reachable remote system, a user may inadvertently cause an NFS mount over X.25 or SLIP, which is unsupported, or through a slow router or gateway. Mounts over slow links may cause excessive retransmissions and degrade performance for all users.

When a user or process requests a directory from an NFS server, AutoFS creates a subdirectory, named after the NFS server, under the local mount point you configured in the automounter master map. (The conventional mount point for the -hosts map is /net.) Then AutoFS mounts all the exported directories from that server under the subdirectory it created. Directories will stay mounted until they are left

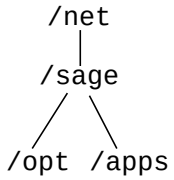
idle for five minutes. The five minute default can be changed by adding the `-t` *duration* option to the `AUTOMOUNT_OPTIONS` variable in the `/etc/rc.config.d/nfsconf` file.

For example, if server `sage` exports `/opt` and `/apps`, and a user on your NFS client types the following command,

```
cd /net/sage/opt/frame
```

the subdirectory `/sage` is created under `/net`, and `/opt` and `/apps` are mounted under `/sage`. Figure 2-4 shows the automounted file structure after the user's command.

Figure 2-4 Automounted Directories from `-hosts` Map—One Server

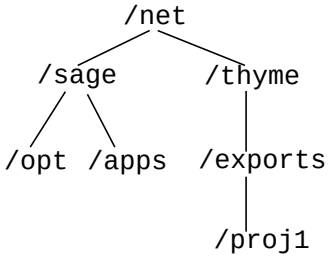


If server `thyme` exports the directory `/exports/proj1`, and a user types the following command,

```
more /net/thyme/exports/proj1/readme
```

the subdirectory `/thyme` is created under `/net`, and `/exports/proj1` is mounted under `/thyme`. Figure 2-5 shows the automounted directory structure after the second user's command.

Figure 2-5 Automounted Directories from `-hosts` Map—Two Servers



The `-hosts` map is an indirect map. It uses the hosts database (the `/etc/hosts` file, the NIS `hosts` map, or BIND [DNS]) to find a host on the network. The Name Service Switch configuration determines which name services will be searched for host information. See "Configuring the Name Service Switch" on page 153.

To Decide Between Direct and Indirect NFS Automounts

- Before you automount a remote directory, decide whether you want to use a direct or indirect automounter map. Table 2-5 lists the advantages and disadvantages of each type of map.

In general, an indirect map is better than a direct map, because it is easier to modify while AutoFS is running, and because it does not cause "mount storms" in directories with many automount points.

However, if your automounted directory must share the same parent directory with local or standard-mounted directories, or if users must always get a complete list of available files and directories when they issue the `ls` command, you should choose a direct map.

Table 2-5 lists the advantages and disadvantages of direct and indirect automounter maps.

Table 2-5 Direct vs. Indirect Automounter Map Types

Direct Map	Indirect Map
Advantage: A user can see the contents of a direct-mounted directory with the `ls` command. If the contents are not currently mounted, `ls` causes them to be mounted.	*Disadvantage:* If a user types `ls` to see the contents of an indirect-mounted directory, it appears empty unless its subdirectories are currently mounted. The user must `cd` to a subdirectory or type `ls` *subdirectory* to cause it to be mounted.
Advantage: Direct-mounted automounted directories can share the same parent directory with local or standard-mounted files and directories.	*Disadvantage:* An indirect map hides any local, standard-mounted, or direct-mounted files or directories underneath the mount point for the map.
Disadvantage: If you add or remove mounts in a direct map, or if you change the local mount point for an existing mount in a direct map, you have to force AutoFS to reread its maps or reboot your system before AutoFS sees the changes you made.	*Advantage:* If you modify an indirect map, AutoFS will see the changes the next time it mounts the directory, so you don't have to force AutoFS to reread its maps.

Direct Map	Indirect Map
Disadvantage: When a user or program accesses a directory containing many direct mount points, all the directories are mounted, whether they are needed or not. This can cause a flurry of mount activity.	*Advantage:* When a user or program accesses a directory containing many indirect mount points, only directories that are already mounted appear.
Disadvantage: When `automount` reads a direct map, it creates an entry for each automounted directory in the internal mount table, `/etc/mnttab`. This can cause the mount table to become very large.	*Advantage:* When `automount` reads an indirect map, it creates only one entry for the entire map in the internal mount table, `/etc/mnttab`. Additional entries are created as directories are actually mounted. The mount table takes up no more space than necessary, because only mounted directories appear in it.

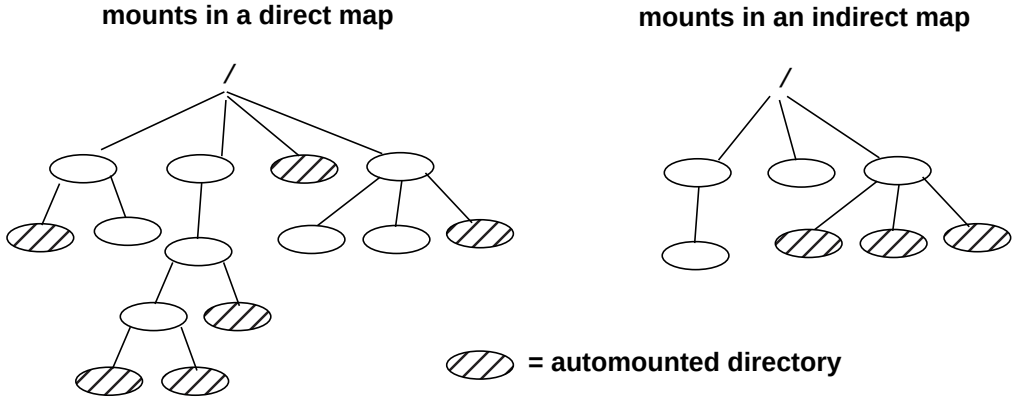
How AutoFS Sets Up Direct and Indirect Mounts

The automounts configured in a direct map may be mounted in various places in the local file system; they do not have to be located under the same parent directory.

The automounts configured in an indirect map are all mounted under the same local parent directory.

Figure 2-6 shows the difference between direct mounts and indirect mounts on an NFS client.

Figure 2-6 The Difference Between Direct Mounts and Indirect Mounts



To Mount a Remote Directory Using a Direct Automounter Map

1. If you are using local files for your automounter maps, use an editor to open or create a direct map in the `/etc` directory. The direct map is commonly called `/etc/auto_direct`. Add a line to the direct map with the following syntax:

   local_directory [*mount_options*] *server*:*remote_directory*

   If you are using NIS to manage your automounter maps, add the line to the direct map on the NIS master server.

2. If you are using local files for your automounter maps, use an editor to open or create the automounter master map in the `/etc` directory. The master map should be called `/etc/auto_master`. If you are using NIS, open the master map on the NIS master server.

   If the direct map you just modified is not listed in the automounter master map, add the following line to the master map:

   `/-` *direct_map_name* [*mount_options*]

3. If you are using NIS to manage your automounter maps, issue the following commands on the NIS master server to rebuild the maps and push them to the slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make auto_master auto_direct
   ```

4. On each host that will use the map you have just modified, issue the following command to force AutoFS to read the modified map:

   ```
   /usr/sbin/automount
   ```

The local directory you configure as the mount point should be empty or non-existent. AutoFS will create any non-existent directories between the root directory and the configured mount point. If the local directory you configure is not empty, any local files or directories in it will be hidden and inaccessible while the remote directory is mounted over it.

CAUTION Do not automount a remote directory on a local directory that is a symbolic link.

If you are using NIS to manage your automounter maps, make sure the local mount point is different from the exported directory on the server. If they are the same, the server may attempt to mount its exported directory over itself, and the directory will become unavailable.

The mount options are the same ones used for standard NFS-mounted directories. See “To Change the Default Mount Options” on page 40 for a list of mount options. The `bg` option cannot be used for an automounted directory. The mount options configured in the direct map override the ones in the master map if there is a conflict.

You can configure all your direct automounts in the same map. Many people use the file name `/etc/auto_direct` for their direct map. If you plan to use NIS to manage your automounter maps, you can have only one direct map in your configuration. If you plan to use NIS to manage your automounter maps, and your file system does not allow file names longer than 14 characters, keep the map name to 10 characters or fewer.

If the direct map name in the automounter master map contains a slash (/), AutoFS assumes it is a local file. If it does not contain a slash, AutoFS uses the Name Service Switch to determine whether it is a file or an NIS map. See “Configuring the Name Service Switch” on page 153.

Before you can mount a remote directory on your system, the remote system where the directory is located must be configured as an NFS server and must export the directory.

You must enable AutoFS before any directories can be automounted. See “To Enable AutoFS” on page 80.

Automounted directories stay mounted until they are left idle for five minutes. The five minute default can be changed by adding the `-t` *duration* option to the `AUTOMOUNT_OPTIONS` variable in the `/etc/rc.config.d/nfsconf` file.

If you change the mount options, the remote server name, or the remote directory name for an existing direct mount while AutoFS is running, the changes you made will take effect the next time the directory is mounted. However, if you change the local directory name in the direct map, or if you change the master map, these changes will not take effect until you issue the `automount` command to force AutoFS to reread its maps.

You can list executable automounter maps in the master map, or include them in local automounter map files. Executable automounter maps return a map entry on standard output when `automountd` supplies them with a key to look up. If they cannot supply a map entry for the key, they should return nothing. AutoFS determines whether a map is executable by checking whether the execute bit is set in its permissions string. If a map is *not* executable, make sure its execute bit is *not* set.

Automounted directories in the `/etc/mnttab` file contain the keyword `ignore` to prevent them from being mounted at boot time.

For more information on AutoFS configuration, type `man 1M automount` at the HP-UX prompt.

Example File Entries for Direct Automounts

Following are example lines from an automounter direct map on NFS client sage. The sharp sign (#) indicates a comment line.

```
# /etc/auto_direct file
# local mount point    mount options  remote server:directory

/auto/project/specs    -nosuid        thyme:/export/project/specs
/auto/project/budget   -nosuid        basil:/export/FY94/proj1
```

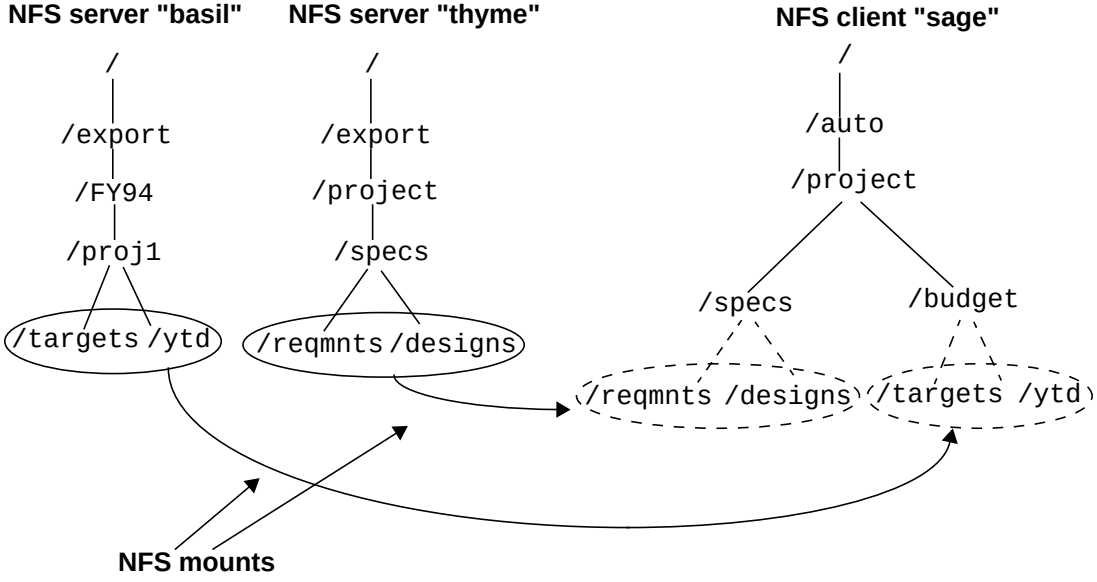
Following are example lines from the automounter master map on NFS client sage.

```
# /etc/auto_master file
# local mount point        map name              mount options

/-                         /etc/auto_direct
```

Figure 2-7 illustrates how the AutoFS sets up the direct mounts for this configuration.

Figure 2-7 Example of Direct Mounts



To Mount a Remote Directory Using an Indirect Automounter Map

1. If you are using local files for your automounter maps, use an editor to open or create an indirect map in the `/etc` directory. Add a line with the following syntax to the indirect map:

   local_subdirectory [*mount_options*] *server*:*remote_directory*

   If you are using NIS to manage your automounter maps, add the line to an indirect map on the NIS master server.

2. If you are using local files for your automounter maps, use an editor to open or create the automounter master map in the `/etc` directory. The master map should be called `/etc/auto_master`. If you are using NIS, open the master map on the NIS master server.

   If the indirect map you just modified is not listed in the automounter master map, add the following line to the master map:

   local_parent_directory indirect_map_name [*mount_options*]

3. If you are using NIS to manage your automounter maps, issue the following commands on the NIS master server to rebuild the maps and push them to the slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make auto_master indirect_mapname
   ```

4. If you modified the automounter master map, issue the following command on each host that will use the map, to force AutoFS to read the modified master map:

   ```
   /usr/sbin/automount
   ```

The *local_subdirectory* specified in the indirect map is the deepest subdirectory in the local directory pathname. For example, if you were mounting a remote directory on `/nfs/apps/draw`, the *local_subdirectory* specified in the indirect map would be `draw`.

The *local_parent_directory* specified in the master map is all but the deepest subdirectory in the local directory pathname. For example, if you were mounting a remote directory on `/nfs/apps/draw`, the *local_parent_directory* specified in the master map would be `/nfs/apps`.

The *local_parent_directory* and *local_subdirectory* should not exist; AutoFS will create them when it mounts the remote directory. If the *local_parent_directory* or *local_subdirectory* contains files or directories, they will be hidden beneath the remote directory when it is mounted.

CAUTION

The *local_subdirectory* and *local_parent_directory* must not be symbolic links.

If you are using NIS to manage your automounter maps, make sure the local mount point is different from the exported directory on the server. If they are the same, the server may attempt to mount its exported directory over itself, and the directory will become unavailable.

The mount options are the same ones used for standard NFS-mounted directories. See "To Change the Default Mount Options" on page 40 for a list of mount options. The `bg` option cannot be used for an automounted directory. The mount options configured in the indirect map override the ones in the master map if there is a conflict.

You can configure indirect automounts in the same indirect map only if their *local_parent_directory*, as specified in the automounter master map, is the same. For example, indirect mounts with the local mount points `/nfs/apps/draw` and `/nfs/apps/word` could be configured in the same indirect map.

Indirect maps are usually called `/etc/auto_`*name*, where *name* is something that helps you remember what is configured in the map. If you plan to use NIS to manage your automounter maps, and if your file system does not support file names longer than 14 characters, keep your indirect map names to 10 characters or fewer.

If the indirect map name in the automounter master map contains a slash (/), AutoFS assumes it is a local file. If it does not contain a slash, AutoFS uses the Name Service Switch to determine whether it is a file or an NIS map. See "Configuring the Name Service Switch" on page 153.

Before you can mount a remote directory on your system, the remote system where the directory is located must be configured as an NFS server and must export the directory.

Automounted directories stay mounted until they are left idle for five minutes. The five minute default can be changed by adding the `-t` *duration* option to the `AUTOMOUNT_OPTIONS` variable in the `/etc/rc.config.d/nfsconf` file.

You must enable AutoFS before any directories can be automounted. See "To Enable AutoFS" on page 80.

If AutoFS is already running when you add an indirect mount to your configuration, you do not have to run the `automount` command unless you change the master map. Any changes you make to an existing

indirect map will take effect the next time AutoFS mounts the directory. However, changes to the master map will not take effect until you issue the `automount` command to force AutoFS to reread its maps.

You can list executable automounter maps in the master map, or include them in local automounter map files. Executable automounter maps return a map entry on standard output when `automountd` supplies them with a key to look up. If they cannot supply a map entry for the key, they should return nothing. AutoFS determines whether a map is executable by checking whether the execute bit is set in its permissions string. If a map is *not* executable, make sure its execute bit is *not* set.

Automounted directories in the `/etc/mnttab` file contain the keyword `ignore` to prevent them from being mounted at boot time.

For more information on AutoFS configuration, type `man 1M automount` at the HP-UX prompt.

Example File Entries for Indirect Automounts

Following are example lines from an automounter indirect map on NFS client `sage`. The sharp sign (#) indicates a comment. Everything from the sharp sign to the end of the line is ignored by AutoFS.

```
# /etc/auto_desktop file
# local mount point       mount options   remote server:directory

draw                     -nosuid          thyme:/export/apps/draw
write                    -nosuid          basil:/exprort/write
```

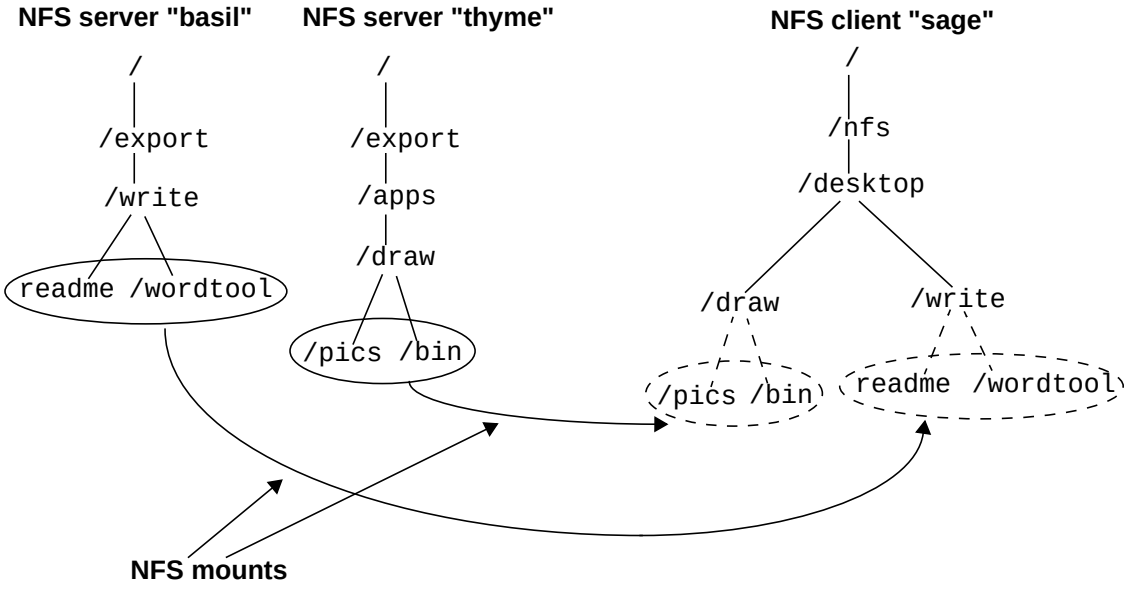
Following are example lines from the automounter master map on NFS client `sage`. The master map also includes an entry for the direct map `/etc/auto_direct`.

```
# /etc/auto_master file
# local mount point         map name           mount options

/-                         /etc/auto_direct
/nfs/desktop               /etc/auto_desktop
```

Figure 2-8 illustrates how AutoFS sets up the indirect mounts for this configuration.

Figure 2-8 How AutoFS Sets Up Indirect Mounts



To Configure Multiple (Replicated) Servers for an Automounted Directory

1. Follow the instructions in "To Mount a Remote Directory Using a Direct Automounter Map" on page 60 or "To Mount a Remote Directory Using an Indirect Automounter Map" on page 64.
2. In the direct or indirect map, modify the line that mounts the remote directory so that multiple servers are listed.

   - If the remote directory has a different name on the different servers, use a syntax like the following example from a direct map:

     ```
     /nfs/proj2/schedule -ro  broccoli:/export/proj2/schedule \
                              cauliflower:/proj2/FY94/schedule
     ```

     AutoFS reads this entry as one line. The line has been broken for readability, and the backslash (\) tells AutoFS that the line continues after the line break.

   - If the remote directory has the same name on every server, use a syntax like the following example from an indirect map:

     ```
     man  -ro  broccoli,cabbage,cauliflower:/usr/share/man
     ```

   - You can assign weights to the various servers, by specifying a number in parentheses after each server name. The lower the weight number, the more likely the server is to be selected.

     ```
     man -ro broccoli(1),cabbage(2),cauliflower(3):/usr/share/man
     ```

     Servers with no weight specified have a default weight of zero (most likely to be selected).

     Server proximity is more important than the weights you assign. A server on the same network segment as the client is more likely to be selected than a server on another network segment, regardless of the weights you assign.

Directories with multiple servers should be mounted read-only to ensure that the versions remain the same on all the servers.

When a user requests access to a directory with multiple servers configured, AutoFS polls all the servers simultaneously and mounts the directory from the server that responds first. Multiple servers give users reliable access to a mounted directory, because if one server is down, the directory can be mounted from another. Also, multiple servers provide

some load balancing across the network; a server that is not busy will respond more quickly to AutoFS's poll than one that is heavily loaded, so the directory will be mounted from the server that is not busy.

If you configure multiple servers on both sides of a gateway, a server on the same side of the gateway as the NFS client will always be used, because it will always respond to the client's poll before the servers on the other side of the gateway.

To Use Environment Variables as Shortcuts in Automounter Maps

1. Use an environment variable anywhere in a direct or indirect automounter map *except* the first field, which specifies the local mount point. An environment variable must be preceded by a dollar sign ($) or enclosed in curly braces {}. The following direct map uses a variable called HOST:

   ```
   /private_files  sage:/export/private_files/$HOST
   ```

2. Add the -D option to the AUTOMOUNTD_OPTIONS variable in the /etc/rc.config.d/nfsconf file to assign a value to the variable, as in the following example:

   ```
   AUTOMOUNTD_OPTIONS="-D HOST='hostname'"
   ```

The example shown above assumes that NFS server sage has subdirectories in its /export/private_files directory that are named after the hosts in its network. Every host in the network can use the same automounter map and the same AUTOMOUNTD_OPTIONS definition to mount its private files from server sage.

For example, when AutoFS starts up on host basil, it assigns the value basil to the HOST variable. Then, when someone requests access to the local /private_files directory on basil, AutoFS mounts /export/private_files/basil from server sage.

Any environment variable that is set to a value may be used in an automounter map. If you do not set the variable with the -D option in /etc/rc.config.d/nfsconf, AutoFS uses the current value of the environment variable on the local host.

You cannot use environment variables in the automounter master map.

To Use Wildcard Characters as Shortcuts in Automounter Maps

1. Use the asterisk (*) in an indirect map as a wildcard character to represent the local subdirectory, when you want the local subdirectory to be the same as the remote system name or the remote subdirectory.
2. Use the ampersand (&) in a direct or indirect map as the remote system name or the remote subdirectory. Whatever is in the local directory name field will replace the ampersand. If you have used an asterisk to represent the local subdirectory, whatever replaces the asterisk (*) in the local subdirectory field also replaces the ampersand (&) in the remote system name or remote subdirectory field.

You cannot use the asterisk (*) wildcard in a direct map.

The following example automounts users' home directories. The home directories are physically located on NFS server `basil`, under the remote directory `/export/home`. On the local NFS client, the home directories will be mounted under `/home`.

Following is the line from the automounter master map `/etc/auto_master` that lists the indirect map `/etc/auto_home`.

```
# /etc/auto_master file
# local mount point       map name            mount options

/home                     /etc/auto_home      nosuid
```

Following is the line from the automounter indirect map `/etc/auto_home` that mounts users' home directories on demand.

```
# /etc/auto_home file
# local mount point      mount options   remote server:directory

*                                        basil:/export/home/&
```

A user's home directory is configured in the `/etc/passwd` file as `/home/`*`username`*. For example, the home directory of user `terry` is `/home/terry`. When Terry logs in, AutoFS looks in the `/etc/auto_home` map and substitutes `terry` for both the asterisk and the ampersand. AutoFS then mounts Terry's home directory from `/export/home/terry` on server `basil` to `/home/terry` on the local NFS client.

The ampersand character can be used to represent both the remote server and the remote subdirectory, in the same line of the indirect map. For example, if users' home directories are physically located on many

different servers, but the directory under which the home directories are located is called /export/home/*servername* on all the servers, the following line in the /etc/auto_home map will mount all users' home directories from any server:

```
*          &:/export/home/&
```

If the home directory of user terry is configured in the /etc/passwd file as /home/basil/terry, when Terry logs in, AutoFS will mount the remote directory /export/home/basil from server basil on the local directory /home/basil.

The line with the asterisk and ampersand should be the last line in an indirect map. AutoFS reads the lines in the indirect map sequentially until it finds a match for the requested local subdirectory. The asterisk (*) matches any subdirectory, so AutoFS stops reading at the line with the asterisk, because it has found a match. Any lines after the asterisk are never read.

For example, if the /etc/auto_home map contains the following lines,

```
*           basil:/export/home/&
charlie     thyme:/export/home/charlie
```

AutoFS attempts to mount /export/home/charlie from host basil. The asterisk is a match for charlie, so AutoFS looks no further and never reads the second line. However, if the /etc/auto_home map contains the following lines,

```
charlie     thyme:/export/home/charlie
*           basil:/export/home/&
```

AutoFS will mount Charlie's home directory from host thyme and everyone else's home directory from host basil.

For more information on AutoFS configuration, type man 1M automount at the HP-UX prompt.

To Automount Users' Home Directories

NOTE

This configuration requires that users' home directories be located under the same directory on all systems in the network. On HP-UX release 9.x or earlier, home directories are usually located under `/users`. On HP-UX release 10.0 or later, home directories are usually located under `/home`. For this reason, you should not set up this configuration until all of your systems are running HP-UX release 10.0 or later.

1. Make sure the machines where users' home directories are located are set up as NFS servers and are exporting the home directories. See "Configuring and Administering an NFS Server" on page 22.
2. In the `/etc/passwd` file on the NFS clients, or in the NIS `passwd` map or NIS+ `passwd` table, configure the home directory of each user as the NFS mount point where the user's home directory will be moutnted. For example, if home directories are mounted under `/home`, Claire's home directory would be configured as `/home/claire` in the `/etc/passwd` file.
3. If you are using local files for your automounter maps, create a file called `/etc/auto_home` on the NFS clients, and add a line to it for each user, like the following example. If you are using NIS to manage your automounter maps, add the lines to the `/etc/auto_home` file on the NIS master server.

   ```
   sammy       thyme:/export/home/&     nosuid
   ```

   The ampersand (&) character takes the value of the user name in each line. In the example above, user `sammy`'s home directory is physically located on host `thyme` in `/export/home/sammy`.
4. If you are using local files for your automounter maps, add the following line to the automounter master map, `/etc/auto_master`, on the NFS clients:

   ```
   /home  /etc/auto_home
   ```

   If you are using NIS to manage your automounter maps, add the line to the `/etc/auto_master` file on the NIS master server.
5. If you are using NIS to manage your automounter maps, issue the following commands on the NIS master server to rebuild the maps and push them to slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make auto_master
   ```

6. Issue the following command, on each NFS client that will use these automounter maps, to force AutoFS to reread the maps:

   ```
   /usr/sbin/automount
   ```

Before you can automount home directories, you must enable AutoFS. See "To Enable AutoFS" on page 80.

Example of Automounting a User's Home Directory

User Howard's home directory is located on NFS server `basil`, where it is called `/export/home/howard`. On all the machines in the network, Howard has the following entry in the `/etc/passwd` file:

```
howard:MILQ3N1tBHXhM:828:Howard:/home/howard:/bin/ksh
```

When Howard logs into any NFS client, AutoFS recognizes `/home` as an AutoFS mount point, because it is configured in the master map:

```
/home  auto_home
```

AutoFS reads the `auto_home` map to find out how to mount Howard's home directory. It finds the following line:

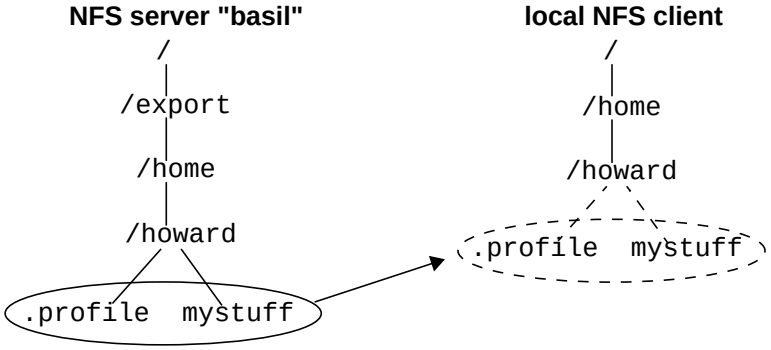
```
howard       basil:/export/home/&     nosuid
```

AutoFS substitutes `howard` for the ampersand (&) character in that line:

```
howard    basil:/export/home/howard   nosuid
```

AutoFS mounts `/export/home/howard` from server `basil` to the local mount point `/home/howard` on the NFS client. Figure 2-9 illustrates this configuration:

Figure 2-9 **Home Directories Automounted with Wildcards**



To Automount Multiple Directories Simultaneously (Hierarchical Mounts)

- Use an editor to create an entry with the following format in a direct or indirect automounter map. (Create the map, if necessary, and add it to the automounter master map.)

```
local_dir    /local_subdirectory [-options] server:remote_directory \
               /local_subdirectory [-options] server:remote_directory \ . . .
```

The backslash (\) characters tell AutoFS to ignore the line breaks, so this entry is effectively all one line.

Map entries with this format cause all the remote directories on the line to be mounted at the same time. For example, the following entry from a direct map mounts the source code and the data files for a project at the same time; whenever anyone requests access to either one, they are both mounted.

```
/our_project   /source   -ro    broccoli:/opt/proj1/src  \
               /datafiles    cauliflower:/opt/proj1/samples/data
```

Because the directories are always mounted simultaneously, you can use relative pathnames to move from one to another, for example,

```
cd ../source
```

Here is another example from an indirect map. In this example, the same mount option (nosuid) applies to all three automounted directories.

```
chap2 -nosuid   /text       sage:/our_book/chap2  \
                /graphics   basil:/our_book/artwork/chap2 \
                /old        sage:/our_book/oldfiles/chap2
```

To Automount a Directory Using CacheFS

Before you mount a file system, you must decide whether to use CacheFS. CacheFS improves read performance for data that will be read more than once. It does not improve write performance at all.

Good choices for cached file systems include man pages and executable programs, which are read multiple times and rarely modified. A bad choice is `/var/mail`, which is modified frequently but is typically read only once and then thrown away.

Follow these steps to automount a directory with CacheFS:

1. On the NFS client host, issue the following command to create a CacheFS directory with the data structures necessary to allow a CacheFS mount:

   `/usr/sbin/cfsadmin -c /`*`cache_directory`*

   For example, if you had a mounted file system called `/disk2`, you could create a CacheFS directory called `/disk2/cache` with the following command:

   ```
   /usr/sbin/cfsadmin -c /disk2/cache
   ```

2. Add a line for the automounted file system to the appropriate automounter direct or indirect map, as in the following examples:

   ```
   # direct map example:
   /usr/dist -ro,nosuid,fstype=cachefs,backfstype=nfs, \
   cachedir=/disk2/cache distserver:/export/dist
   ```

   ```
   # indirect map example:
   proj1  -nosuid,fstype=cachefs,backfstype=nfs,\
                     cachedir=/disk2/cache \
          /src  testbox1:/export/proj1/src
          /data testbox2:/export/proj1/data
   ```

3. If you modified a direct map or the automounter master map, issue the following command, on each NFS client that will use the map, to force AutoFS to reread its maps:

   ```
   /usr/sbin/automount
   ```

You can specify caching in an NIS automounter map only if all clients who will use the map have their caching directory set up in the same location (`/disk2/cache`, in the examples).

For more informationon CacheFS, see Chapter 3, “Configuring the Cache File System (CacheFS),” on page 95.

To Include an Automounter Map in Another Automounter Map

- To include the contents of an automounter map in another automounter map, add a plus sign (+) before the map name, as in the following example:

```
# /etc/auto_home file
# local mount point    mount options    remote server:directory

basil                  -nosuid          basil:/export/home/basil
+auto_home
```

Assume the `/etc/auto_home` map is listed in the master map with the following line:

```
/home           /etc/auto_home
```

This example has the following effect:

If a user logs in whose home directory is in `/home/basil`, AutoFS will mount the directory `/export/home/basil` from host `basil`.

If a user logs in whose home directroy is in `/home/sage`, `/home/thyme`, or any subdirectory of `/home` other than `basil`, AutoFS will consult the NIS map `auto_home` for information on mounting the user's home directory.

The plus sign (+) tells AutoFS to look in a different map for the information it needs to mount the directory. If the map name following the plus sign begins with a slash, AutoFS assumes it is a local file. If the map name contains no slashes, AutoFS uses the Name Service Switch to determine whether it is a file or an NIS map. See "Configuring the Name Service Switch" on page 153.

You can include an automounter map inside a local file but not inside an NIS map.

For more information, type `man 1M automount` or `man 4 nsswitch.conf`.

To Create a Hierarchy of Automounter Maps

An organization made up of many departments may wish to organize a shared automounted directory structure. In the following example, the shared top-level directory is called `/org`. The `/org` directory contains several subdirectories, listed in the `auto_org` automounter map. Each department administers its own automounter map for its subdirectory.

The automounter master map needs just a single entry for `/org`:

```
# auto_master map
# Directory           Map Name

/org                  auto_org
```

The `auto_org` map looks like this:

```
finance    -fstype=autofs    auto_finance
marketing  -fstype=autofs    auto_marketing
legal       -fstype=autofs    auto_legal
research     -fstype=autofs    auto_research
eng        -fstype=autofs    auto_eng
```

And the engineering department's map, `auto_eng`, looks like this:

```
releases                      bigiron:/export/releases
tools                         mickey,minnie:/export/tools
source      -fstype=autofs    auto_eng_source
projects    -fstype=autofs    auto_eng_projects
```

A user in the "blackhole" project within engineering might use the following path:

```
/org/eng/projects/blackhole
```

Beginning with the AutoFS mount at `/org`, the evaluation of this path would dynamically create additional AutoFS mounts at `/org/eng` and `/org/eng/projects`. Since AutoFS mounts are created only when needed, changes to maps require no action to become visible at the user's workstation. The `automount` command needs to be run only when changes are made to the master map or to a direct map.

Hierarchical automounter maps provide a framework within which large shared filesystems can be organized. Together with NIS, which allows you to share information across administrative domains, the maintenance of the shared namespace can be effectively decentralized.

To Turn Off an Automounter Map with the `-null` Map

1. Add a line with the following syntax to the automounter master map:

   local_directory `-null`

2. If AutoFS is running, issue the following command, on each client that will use the map, to force AutoFS to reread its maps:

   `/usr/sbin/automount`

The `-null` option "turns off" the map that is mounted on *local_directory*. For example, if the NIS `auto_master` map mounts the `auto_home` map on `/home`, and you include the following line in your local `/etc/auto_master` file,

```
/home -null
```

the NIS `auto_home` map will not be used on your system.

The `-null` option is useful for turning off NIS automounter maps that do not apply to your host.

You can also replace NIS maps with local maps, as in the following example from `/etc/auto_master`:

```
/home /etc/auto_ourhome
```

Because AutoFS reads the local `/etc/auto_master` file before the NIS `auto_master` map, this entry causes AutoFS to look for mount information in the local file `/etc/auto_ourhome` instead of the `auto_home` NIS map.

For more information, type `man 1M automount`.

To Enable AutoFS

1. In the `/etc/rc.config.d/nfsconf` file, make sure the `NFS_CLIENT` and `AUTOMOUNT` variables are set to 1, as follows:

   ```
   NFS_CLIENT=1
   AUTOMOUNT=1
   ```

2. Issue the following command to run the NFS client startup script:

   ```
   /sbin/init.d/nfs.client start
   ```

   or

   ```
   /sbin/init.d/autofs start
   ```

The `nfs.client start` script will start any NFS client processes that are not already running, including AutoFS. If you want to start *only* AutoFS, use the `autofs start` script.

When AutoFS starts up, it uses the Name Service Switch to determine which name services you are using and to find the master maps that are available from those name services.

For more information, type `man 4 nsswitch.conf` or `man 1M automount` at the HP-UX prompt.

To Disable AutoFS

1. In the `/etc/rc.config.d/nfsconf` file, make sure the `NFS_CLIENT` and `AUTOMOUNT` variables are set to 1, as follows:

   ```
   NFS_CLIENT=1
   AUTOMOUNT=1
   ```

2. Issue the following command to run the AutoFS shutdown script:

   ```
   /sbin/init.d/autofs stop
   ```

CAUTION Do not kill the `automound` daemon with the `kill` command. It does not die gracefully. It does not unmount AutoFS mount points before it dies. Use the `autofs stop` script to ensure that `automound` dies cleanly.

To Verify Your AutoFS Configuration

1. Type the following command to change the current working directory to an automounted directory:

   `/usr/bin/cd` *local_directory*

   where *local_directory* is the configured mount point in the automounter map.

2. Type the following command to verify that the contents of the remote directory have been mounted under the local mount point:

   `/usr/bin/ls`

If the directory is configured in an indirect map, issuing the `ls` command from the parent directory will display nothing. When you `cd` to a subdirectory configured in the indirect map, or issue the command `ls` *subdirectory*, the subdirectory will be mounted.

Therefore, if you have the following indirect map configuration,

```
# /etc/auto_master file
# local mount point         map name             mount options

/nfs/desktop                /etc/auto_desktop


# /etc/auto_desktop file
# local mount point        mount options   remote server:directory

draw                      -nosuid          thyme:/export/apps/draw
write                     -nosuid          basil:/export/write
```

and you issue the following commands,

```
cd /nfs/desktop
ls
```

the `ls` command will produce no output, because the `draw` and `write` subdirectories are not currently mounted. However, if you issue the following commands,

```
cd /nfs/desktop/write
cd /nfs/desktop/draw
cd ..
ls
```

the `ls` command will display

```
draw         write
```

If AutoFS is not mounting your configured directories, see "Troubleshooting NFS Services" on page 173.

To Modify or Remove (Unmount) an Automounted Directory

1. If you are planning to remove an automounted directory, issue the following command to determine whether the directory is currently in use:

   `/usr/sbin/fuser -cu` *local_mount_point*

   This command lists the process IDs and user names of everyone using the mounted directory.

2. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can issue the following command to kill all the processes using the mounted directory:

   `/usr/sbin/fuser -ck` *local_mount_point*

3. Use an editor to make your changes to the direct or indirect map.

4. If you removed the last entry in the direct or indirect map, remove the line for that map in the automounter master map.

5. If you made any changes to the master map, or if you added or modified a local mount point in a direct map, run the following command to force AutoFS to reread its maps:

   `/usr/sbin/automount`

Configuring and Using NFS Netgroups

This section tells you how to create and use NFS netgroups to restrict NFS access to your system. It describes the following tasks:

- To Create Netgroups in the /etc/netgroup File
- To Use Netgroups in Configuration Files

To Create Netgroups in the /etc/netgroup File

1. If you are using the local /etc/netgroup file or the NIS netgroup map for netgroups, add lines with the following syntax to the /etc/netgroup file. If you are using NIS, be sure to edit the /etc/netgroup file only on the NIS master server.

   netgroup_name (*host*, *user*, *NIS_domain*), (*host*, *user*, *NIS_domain*) ...

2. If you are using NIS to manage your netgroups database, issue the following command on the NIS master server to generate the netgroup, netgroup.byhost, and netgroup.byuser maps from the /etc/netgroup file and push the generated maps out to the NIS slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make netgroup
   ```

A netgroup can be used in most NFS and NIS configuration files instead of a host name or a user name. A netgroup does not create a relationship between users and hosts. When a netgroup is used in a configuration file, it represents either a group of hosts or a group of users but never both.

If you are using BIND (DNS) for hostname resolution, hosts must be specified as fully qualified domain names, for example turtle.bio.nmt.edu.

If the *host*, *user*, or *NIS_domain* is left blank in a netgroup, that field can take any value. If a dash (-) is specified in any field of a netgroup, that field can take no value.

The *NIS_domain* field specifies the NIS domain in which the (*host*, *user*, *NIS_domain*) triple is valid. For example, if the netgroup database contains the following netgroup,

```
myfriends (sage,-,bldg1), (cauliflower,-,bldg2), (pear,-,bldg3)
```

and an NFS server running NIS in the domain bldg1 exports a directory only to the netgroup myfriends, only host sage may mount that directory. The other two triples are ignored, because they are not valid in the bldg1 domain.

If an HP-UX host not running NIS exports a directory to the netgroup myfriends, the *NIS_domain* field is ignored, and all three hosts (sage, cauliflower, and pear) may mount the directory.

If the netgroup database contains the following netgroup,

```
mydomain (,,bldg1)
```

and a host in the NIS domain `bldg1` exports a directory to the netgroup `mydomain`, any host *in any domain* may mount the directory, because the *host* field is blank.

If an HP-UX host not running NIS exports a directory to the netgroup `mydomain`, shown above, the *NIS_domain* field is ignored, but the *host* field is used, so any host *in any domain* may mount the directory.

If a host in the NIS domain `bldg2` exports a directory to the netgroup `mydomain`, no host in any domain may mount the directory, because the triple is not valid in the `bldg2` domain, so it is ignored.

Netgroup Examples

The following netgroup specifies a group of hosts:

```
trusted_hosts (sage, , ), (basil, , ), (thyme, , )
```

The `trusted_hosts` netgroup could be used in the `-access` option of a line in the `/etc/exports` file, as follows:

```
/usr -access=trusted_hosts
```

The following netgroup specifies a group of users:

```
administrators ( ,jane, ), ( ,art, ), ( ,mel, )
```

If this netgroup were ever accidentally included in a list of hosts rather than users, the blank space would be interpreted as a wildcard meaning any host. For example, if someone used this netgroup in a `-access` list in the `/etc/exports` file, any host would have access to the exported directory. For this reason, if a netgroup is used strictly as a list of users, it is better to put a dash in the host field, as follows:

```
administrators (-,jane, ), (-,art, ), (-,mel, )
```

The dash indicates that no hosts are included in the netgroup.

The `trusted_hosts` and `administrators` netgroups could be used together in the `/etc/hosts.equiv` file, as follows:

```
+@trusted_hosts  +@administrators
```

The first netgroup would be read for host names, and the second would be read for user names. Users in the `administrators` netgroup could log into the local host from any host in the `trusted_hosts` netgroup without supplying a password.

The two netgroups could be combined into one, as follows:

```
goodguys (sage,jane, ), (basil,art, ), (thyme,mel, )
```

If the two netgroups were combined this way, the same netgroup could be used as both the host name and the user name in the `/etc/hosts.equiv` file:

```
+@goodguys    +@goodguys
```

The first occurrence of it would be read for the host name, and the second occurrence would be read for the user name. No relationship exists between the host and user in any of the triples. For example, user `jane` might not even have an account on host `sage`.

A netgroup can contain other netgroups, as in the following example:

```
root-users (dill,-,  ), (sage,-, ), (thyme,- , ), (basil,-, )
mail-users (rosemary, , ), (oregano, , ), root-users
```

The `root-users` netgroup is a group of four systems. The `mail-users` netgroup uses the `root-users` netgroup as part of a larger group of systems. The blank space in the third field of each triple indicates that these netgroups are valid in any NIS domain.

To Use Netgroups in Configuration Files

Netgroups may be used in the following files:

- `/etc/exports`, in the `-access` list
- `/etc/hosts.equiv` or `$HOME/.rhosts`, in place of a host name or user name
- `/etc/passwd`, to tell processes whether to look in the NIS password database for information about the users in the netgroup
- `/etc/group`, to tell processes whether to look in the NIS group database for information about the users in the netgroup

The next few sections explain how to use netgroups in these files.

Using Netgroups in the `/etc/exports` File

In the `/etc/exports` file, netgroups can be used in the list of NFS clients following the `-access` option, as in the following example:

```
/var/mail -access=mail_clients
```

The `mail_clients` netgroup is defined as follows:

```
mail_clients (cauliflower, , ), (broccoli, , ), (cabbage, , )
```

Only the host names from the netgroup are used. If the netgroup also contains user names, these are ignored. This netgroup is valid in any NIS domain, because the third field in each triple is left blank.

Using Netgroups in the `/etc/hosts.equiv` or `$HOME/.rhosts` File

In the `/etc/hosts.equiv` file, or in a `.rhosts` file in a user's home directory, netgroups can be used in either the host name field or the user name field, as in the following example:

```
+@our_friends    +@our_friends
```

The netgroup `our_friends` can be used as both the host name and the user name, because it includes both host names and user names, as follows:

```
our_friends (sage,sara, ), (sage,eric, ), (dill,-, ), ( ,monica, )
```

The blank host name field in the fourth triple serves as a wildcard, allowing users from any host on the network to log in without supplying a password. However, only the users listed in the netgroup are given this privileged access, because each user name field contains either a user name or a dash.

Netgroups can also be used to deny privileged access to certain hosts or users in the `/etc/hosts.equiv` or `$HOME/.rhosts` file, as in the following example,

```
+   -@vandals
```

The plus sign (+) is a wildcard in the `/etc/hosts.equiv` or `$HOME/.rhosts` file syntax, allowing privileged access from any host in the network. The netgroup `vandals` is defined as follows:

```
vandals ( ,pat, ), ( ,harriet, ), ( ,reed, )
```

All users *except* those listed in the `vandals` netgroup can log into the local system without supplying a password from any system in the network.

CAUTION Any users who are denied privileged access in the `/etc/hosts.equiv` file can still be allowed privileged access in a user's `$HOME/.rhosts` file. The `$HOME/.rhosts` file is read after the `/etc/hosts.equiv` file and overrides it.

For more information, type `man 4 hosts.equiv` at the HP-UX prompt.

Using Netgroups in the /etc/passwd File

In the `/etc/passwd` file, netgroups can be used to indicate whether user information should be looked up in the NIS `passwd` database.

The following example line from the `/etc/passwd` file indicates that users in the netgroup `animals` should be looked up in the NIS `passwd` database:

```
+@animals
```

The `animals` netgroup is defined as follows in the `/etc/netgroup` file:

```
animals (-,mickey, ), (-,daffy, ), (-,porky, ), (-,bugs, )
```

Note that the `/etc/passwd` file is searched sequentially, so if user `mickey`, `daffy`, `porky`, or `bugs` appears before the `animals` netgroup in the `/etc/passwd` file, the NIS database will never be consulted for information on that user.

The Name Service Switch configuration is used to determine where to look for the contents of a netgroup. See "Configuring the Name Service Switch" on page 153.

Netgroups can also be used to prevent lookups of certain users in the NIS `passwd` database. The following example lines from the `/etc/passwd` file indicate that if the NIS `passwd` database contains entries for users in the `bears` netgroup, these entries cannot be used on the local system. Any other users can be looked up in the NIS database.

```
-@bears
+::-2:60001:::
```

The line beginning with + causes the NIS database to be searched for any users (except those in the `bears` netgroup) who are not listed before the line beginning with +.

For more information on NIS, see "Configuring and Administering NIS" on page 101.

For information on the `/etc/passwd` file, type `man 4 passwd` at the HP-UX prompt.

Using Netgroups in the /etc/group File

In the `/etc/group` file, netgroups can be used to indicate whether group information about certain users should be looked up in the NIS `group` database.

The following example line from the `/etc/group` file indicates that group information for users in the netgroup `animals` can be found in the NIS `group` database:

```
+@animals
```

The `animals` netgroup is defined as follows in the `/etc/netgroup` file:

```
animals (-,mickey, ), (-,daffy, ), (-,porky, ), (-,bugs, )
```

Members of the `animals` netgroup can belong to groups listed in the local `/etc/group` file as well as in the NIS `group` database. The following lines in the `/etc/group` file give users `bugs` and `daffy` membership in the group `wiseguys` and in any group in the NIS database that includes them as members:

```
wiseguys::22:bugs,daffy
+@animals
```

Netgroups can also be used in the `/etc/group` file to prevent lookups for certain users. The `bears` netgroup is defined as follows in the `/etc/netgroup` file:

```
bears (-,yogi, ), (-,smokey, ), (-,pooh, )
```

The following lines in the `/etc/group` file allow user `pooh` membership in group `teddybears` but not in any other group listed in the NIS database or after the `-@bears` line in the `/etc/group` file:

```
teddybears::23:pooh,paddington
-@bears
```

For more information on NIS, see “Configuring and Administering NIS” on page 101.

For information on the `/etc/group` file, type `man 4 group` at the HP-UX prompt.

Configuring the Other NFS Daemons and Services

If you want to use some of the other NFS services, like the Remote Execution Facility (REX) or the `rup`(1) and `rusers`(1) commands, this section tells you how to enable those daemons and services. This section tells you how to perform the following tasks:

- To Enable the Other NFS Services
- To Restrict Access to the Other NFS Services

To Enable the Other NFS Services

1. In the `/etc/inetd.conf` file, use a text editor to uncomment the lines that begin with "`rpc`." (Delete the sharp sign [#] in the first column.)

   If the lines do not exist, type them into the `/etc/inetd.conf` file. Table 2-6 gives the line you need to enter for each NFS service.

2. If NFS is not yet running on your system, issue the following command:

   ```
   /sbin/init.d/nfs.client start
   ```

3. Issue the following command to force `inetd` to read its configuration file:

   ```
   /usr/sbin/inetd -c
   ```

CAUTION Do not issue the `/usr/sbin/inetd` command if NFS is not yet running on your system. The NFS startup script starts the `portmap`(1M) process, which *must* be running before you start `inetd`.

Table 2-6 lists the NFS daemons and services that can be started by the `inetd` daemon. It briefly describes each one and tells you which man pages you can read for more information. It also gives the line that configures each service in the `inetd.conf` file.

You cannot use SAM to enable the other NFS services.

Table 2-6 Other NFS Services

`rexd`	The `rpc.rexd` program is the server for the `on` command, which starts the Remote Execution Facility (REX). The `on` command sends a command to be executed on a remote system. The `rpc.rexd` program on the remote system executes the command, simulating the environment of the user who issued the `on` command. See "Configuring and Using the Remote Execution Facility (REX)" on page 165, or see man pages `rexd`(1M) and `on`(1). The following line configures `rexd` in `inetd.conf`: `rpc stream tcp nowait root /usr/sbin/rpc.rexd 100017 1 rpc.rexd`

`rstatd`	The `rpc.rstatd` program answers requests from the `rup` command, which collects and displays status information about the machines on the local network. For more information, see man pages `rstatd`(1M) and `rup`(1). The following line configures `rstatd` in `inetd.conf`: `rpc dgram udp wait root /usr/lib/netsvc/rstat/rpc.rstatd 100001 1-3 \ rpc.rstatd`
`rusersd`	The `rpc.rusersd` program responds to requests from the `rusers` command, which collects and displays information about all users logged into the machines on the local network. For more information, see man pages `rusersd`(1M) and `rusers`(1). The following line configures `rusersd` in `inetd.conf`: `rpc dgram udp wait root /usr/lib/netsvc/rusers/rpc.rusersd 100002 1-2 \ rpc.rusersd`
`rwalld`	The `rpc.rwalld` program handles requests from the `rwall` program. The `rwall` program sends a message to a specified machine where the `rpc.rwalld` program is running, and the message is written to all users logged onto the machine. For more information, see man pages `rwalld`(1M) and `rwall`(1M). The following line configures `rwalld` in `inetd.conf`: `rpc dgram udp wait root /usr/lib/netsvc/rwall/rpc.rwalld 100008 1 \ rpc.rwalld`
`sprayd`	The `rpc.sprayd` program is the server for the `spray` command, which sends a stream of packets to a specified host and then reports how many were received and how fast. For more information, see man pages `sprayd`(1M) and `spray`(1M). The following line configures `sprayd` in `inetd.conf`: `rpc dgram udp wait root /usr/lib/netsvc/spray/rpc.sprayd 100012 1 \ rpc.sprayd`
`rquotad`	The `rpc.rquotad` program responds to requests from the `quota` command, which displays information about a user's disk usage and limits. For more information, see man pages `rquotad`(1M) and `quota`(1). The following line configures `rquotad` in `inetd.conf`: `rpc dgram udp wait root /usr/sbin/rpc.rquotad 100011 1 rpc.rquotad`

To Restrict Access to the Other NFS Services

- In the `/var/adm/inetd.sec` file, create a line with the following syntax for each service to which you want to restrict access:

  service {`allow`} *host_or_network* [*host_or_network*`...`]
  {`deny`}

If the `/var/adm/inetd.sec` file does not exist, you will have to create it.

service must match one of the service names in the `/etc/rpc` file.

Specify either `allow` or `deny` but not both. Enter only one line per service.

host_or_network can be either an official host name or network name or an IP address. Any of the four numbers in an IP address can be specified as a range (for example, 1-28) or the wildcard character (*).

The `inetd.sec` file is checked only when the service is started. If a service remains active and accepts more requests without being restarted, the `inetd.sec` file is not checked again.

You can use SAM to modify the `/var/adm/inetd.sec` file.

For more information see the man pages `inetd.conf`(4) and `inetd.sec`(4).

Examples from `/var/adm/inetd.sec`

The following example allows only hosts on subnets 15.13.2.0 through 15.13.12.0 to use the `spray` command:

```
sprayd allow 15.13.2-12.0
```

The following example prevents host `cauliflower` from using the `rwall` command:

```
rwalld deny cauliflower
```

3 Configuring the Cache File System (CacheFS)

The Cache File System (CacheFS), is a general purpose file system caching mechanism that improves NFS server performance and scalability by reducing server and network load. CacheFS provides the ability to cache one file system on another. In an NFS environment, CacheFS increases the client per server ratio, reduces server and network loads, and improves performance for clients on slow links (for example, PPP).

CacheFS performs local disk caching of file systems, which reduces the network traffic. Individual client machines become less reliant on the server, thereby decreasing overall server load, which leads to an increase in server performance.

By default, CacheFS maintains consistency with the back file system using a consistency checking model like that of NFS (polling for changes in file attributes).

Following are some CacheFS terms that will be used in this chapter:

back file system	The file system that is being cached. On HP-UX 10.20, NFS is the only supported back file system.
front file system	The file system that contains the cached data. On HP-UX 10.20, HFS and JFS are the supported front file systems.
cold cache	A cache that does not yet have any data in its front file system. In this case, requested data must be copied from the back file system to the front file system (that is, the cache must be populated). An attempt to reference data that is not yet cached is called a “cache miss.”
warm cache	A cache that contains the desired data in its front file system. In this case, the cached data can be returned to the user without requiring any action from the back file system. An attempt to reference data that has been cached is called a “cache hit.”

Configuring CacheFS

Before you mount a file system, you must decide whether to use CacheFS. CacheFS improves read performance for data that will be read more than once. It does not improve write performance at all.

The first time data is read from an NFS-mounted file system, there is actually some overhead while CacheFS writes the data to its local cache. After the data is written to the cache, read performance for the file system is significantly improved.

Good choices for cached file systems include man pages and executable programs, which are read multiple times and rarely modified. A bad choice is `/var/mail`, which is modified frequently but is typically read only once and then thrown away.

You cannot use SAM to mount a file system with CacheFS.

You can use CacheFS to cache NFS-mounted or automounted NFS file systems. Before you can mount a file system using CacheFS, you must configure a local file system as the cache directory.

This section gives instructions for completing the following tasks:

- To Configure a Local File System as Cache
- To Mount an NFS File System Using CacheFS
- To Automount a File System Using CacheFS

For more information on CacheFS, see the following man pages: `cfsadmin`(1M), `fsck_cachefs`(1M), `mount`(1M), `mount_cachefs`(1M), and `cachefsstat`(1M).

To Configure a Local File System as Cache

1. If necessary, configure and mount an HFS or JFS file system on the client system where data will be cached. See the *HP-UX System Administration Tasks* manual for more information.

   No special disk partitioning is necessary for creating a CacheFS front file system. If you already have a mounted file system with sufficient disk space for caching your NFS file systems, you can create a subdirectory in the existing file system to use for your CacheFS front file system.

2. Issue the following command to create a CacheFS directory with the data structures necessary to allow a CacheFS mount:

   `/usr/sbin/cfsadmin -c /`*`cache_directory`*

   For example, if you had a mounted file system called `/disk2`, you could create a CacheFS directory called `/disk2/cache` with the following command:

   ```
   /usr/sbin/cfsadmin -c /disk2/cache
   ```

CacheFS manages its resources most effectively in cases where the entire front file system is dedicated to caching, or in cases where the non-cache portions of the front file system are static, read-only files.

CacheFS allows more than one file system to be cached in the same cache. There is no need to create a separate cache directory for each CacheFS mount. In typical usage, you need to run `cfsadmin -c` only once to create a single cache for all of your CacheFS mounts.

For more information, type `man 1M cfsadmin` at the HP-UX prompt.

To Mount an NFS File System Using CacheFS

Before you can mount an NFS file system with CacheFS, you must configure a directory in a local file system as cache. See "To Configure a Local File System as Cache" on page 98.

1. Issue the mount(1M) command to mount an NFS file system using CacheFS, as in the following examples:

   ```
   mount -F cachefs -o backfstype=nfs,cachedir=/disk2/cache \
                   nfsserver:/opt/frame /opt/frame
   ```

2. Add a line to the `/etc/fstab` file, as in the following example, to cause your NFS file system to be mounted at system boot:

   ```
   nfsserver:/opt/frame /opt/frame cachefs \
   backfstype=nfs,cachedir=/disk2/cache 0 0
   ```

This example NFS-mounts the directory `/opt/frame` from server `nfsserver` to the local `/opt/frame` directory. Now, `/opt/frame` can be accessed just like any mounted file system. As data in `/opt/frame` is referenced, it will be copied into `/disk2/cache`. Further references to the data will access the data on the local disk instead of the data on the remote server.

For more information, type `man 1M mount` at the HP-UX prompt.

To Automount a File System Using CacheFS

Before you can automount an NFS file system with CacheFS, you must configure a directory in a local file system as cache. See "To Configure a Local File System as Cache" on page 98.

1. Add a line for the automounted file system to the appropriate automounter direct or indirect map, as in the following examples:

```
# direct map example:
/usr/dist -ro,nosuid,fstype=cachefs,backfstype=nfs, \
cachedir=/disk2/cache distserver:/export/dist

# indirect map example:
proj1  -nosuid,fstype=cachefs,backfstype=nfs,\
                    cachedir=/disk2/cache \
       /src  testbox1:/export/proj1/src
       /data testbox2:/export/proj1/data
```

2. If you modified a direct map or the automounter master map, issue the following command, on each NFS client that will use the map, to force AutoFS to reread its maps:

```
/usr/sbin/automount
```

You can specify caching in an NIS automounter map only if all clients who will use the map have their caching directory set up in the same location (`/disk2/cache`, in the examples).

For more information, type `man 1M automount` at the HP-UX prompt.

4 Configuring and Administering NIS

The Network Information Service (NIS), previously called “Yellow Pages,” is a distributed database system that allows you to maintain commonly used configuration information on a master server and propagate the information to all the hosts in your network. This chapter explains how to configure and administer the servers and clients in an NIS domain. It contains the following sections:

- Overview of NIS
- Planning the NIS Network
- Configuring and Administering an NIS Master Server
- Configuring and Administering an NIS Slave Server
- Configuring and Administering an NIS Client
- Configuring and Administering Secure RPC
- Summary of NIS Commands

NOTE NIS is not supported across extended LANs (LANs separated by routers or bridges). NIS is also not supported across WAN links, like X.25 and SLIP.

Overview of NIS

NIS allows you to administer the configuration of many hosts from a central location. Common configuration information, which would have to be maintained separately on each host in a network without NIS, can be stored and maintained in a central location and propagated to all of the nodes in the network.

Information Managed by NIS

By default, NIS manages the following configuration files:

- `/etc/hosts`, a file that maps internet addresses to host names.
- `/etc/passwd`, a list of the users on your system, along with their passwords, home directories, and other information.
- `/etc/group`, a list of groups of users.
- `/etc/netgroup`, a list of NFS netgroups, which are groups of host names or user names used for allowing or denying access to systems and services.
- `/etc/services`, a file that associates network services with their port numbers and protocols.
- `/etc/protocols`, a file that associates network protocols with protocol numbers.
- `/etc/networks`, a list of network names and numbers.
- `/etc/rpc`, a file that maps RPC program names to program numbers.
- `/etc/auto_master`, an NFS automounter map that lists the direct and indirect automounter maps and their mount points.
- `/etc/mail/aliases`, a list of `sendmail` aliases.
- `/etc/publickey`, a list of secure RPC encryption keys.
- `/etc/netid`, a list of secure RPC netnames (`unix.`*UID*`@`*domainname* or `unix.`*hostname*`@`*domainname*) for users and hosts outside your NIS domain.

- `/etc/vhe_list`, a configuration file for the Virtual Home Environment. (Type `man 4 vhe_list` for more information.) VHE is not supported on 10.0 and later releases.

The information in these files is put into NIS databases automatically when you create an NIS master server. Other system files may be managed by NIS, if you wish to customize your configuration.

Structure of the NIS Network

The center of the NIS network is the **NIS master server**. When you create an NIS master server, the configuration files on that host are used to create **NIS maps**, which are hashed database versions of the configuration files. Once the NIS network is set up, any changes to the maps must be made on the master server.

In addition to the master server, you can create backup servers, called **NIS slave servers**, to take some load off the master server and to substitute for the master server when it is down. When you create an NIS slave server, the maps on the master server are transferred to the slave server. Whenever a change is made to a map on the master server, the modified map must be transferred to the slave servers.

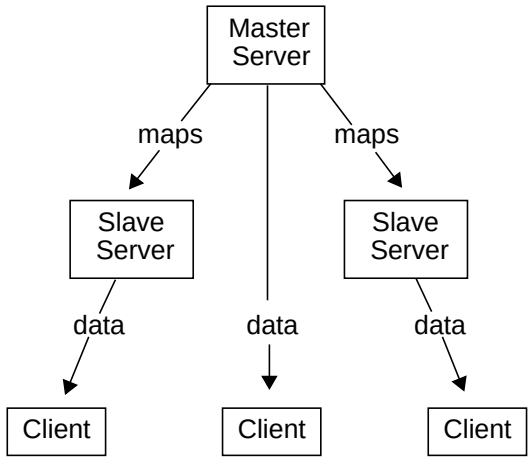
Typically, all the hosts in the network, including the master and slave servers, are **NIS clients**. Whenever a process on an NIS client requests configuration information, it calls NIS instead of looking in its local configuration files. (For group and password information and mail aliases, the `/etc` files may be consulted first, and NIS may be consulted if the requested information is not found in the `/etc` files.)

The set of maps shared by the servers and clients is called the **NIS domain**. The master copies of the maps are located on the NIS master server, in the directory `/var/yp/`*domainname*. Under the *domainname* directory, each map is stored as two files: *mapname*`.dir` and *mapname*`.pag`. Each slave server has an identical directory containing the same set of maps.

When a client starts up, it broadcasts a request for a server that serves its domain. Any server that has the set of maps for the client's domain may answer the request. The client "binds" to the first server to answer its request, and that server answers all of its NIS queries.

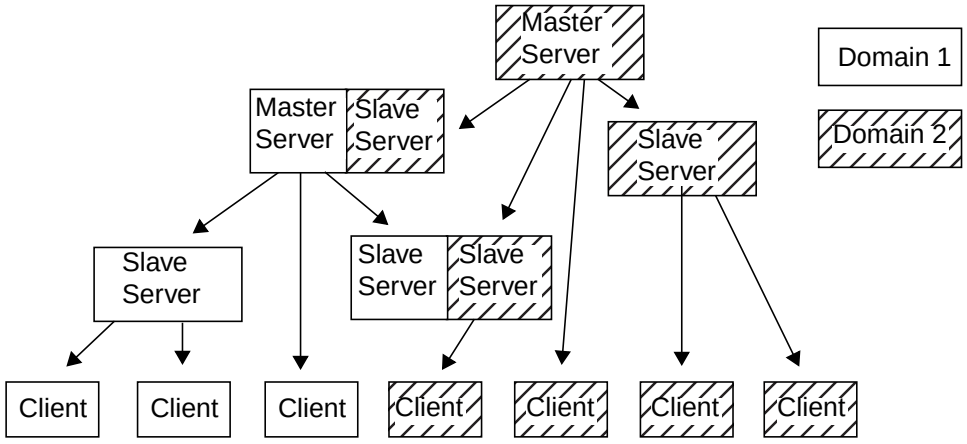
Figure 4-1 shows the flow of information in an NIS domain.

Figure 4-1 Flow of Information in an NIS Network



A host cannot be the master server for more than one NIS domain. However, a master server for one domain may be a slave server for another domain. A host can be a slave server for multiple domains. A client belongs to only one domain. Figure 4-2 shows an NIS network with servers that serve multiple domains.

Figure 4-2 Servers that Server Multiple NIS Domains



Planning the NIS Network

This section explains how to plan the layout of your NIS network. It tells you how to perform the following tasks:

- To Determine the Number of NIS Domains You Need
- To Determine the Number of NIS Servers You Need
- To Determine Which Hosts Will Be NIS Servers
- To Draw an NIS Network Map

To Determine the Number of NIS Domains You Need

For many sites, all hosts can belong to the same domain, and it is not necessary to set up more than one. However, you might want to create multiple domains for the following reasons:

- If your site is divided into multiple administrative departments, with a different system administrator for each department, you should allow each system administrator to maintain a separate NIS domain.
- If your site is divided into multiple administrative departments, and each department requires different configuration data and allows access to different users and hosts, you should create a separate NIS domain for each administrative department.

To Determine the Number of NIS Servers You Need

Following are some guidelines for determining the number of NIS servers you will need in your domain:

- You must put a server on each subnetwork in your domain. When a client starts up, it broadcasts a message to find the nearest server. This broadcast message is not propagated across routers or gateways, so each subnet must have at least one server.
- In general, a server can serve about 30 NIS clients if the clients and servers run at the same speed. If the clients are faster than the servers, you will need more servers. If the clients are slower than the servers, each server can serve 50 or more clients.

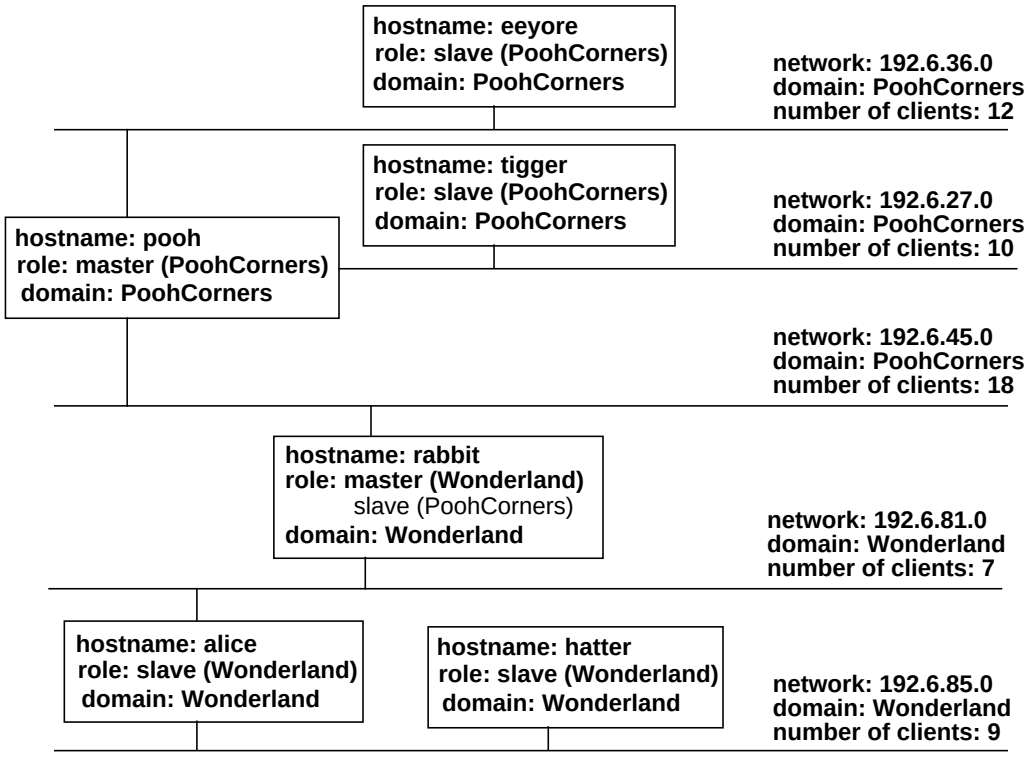
To Determine Which Hosts Will Be NIS Servers

- Choose servers that are reliable and highly available.
- Choose fast servers that are not used for CPU-intensive applications. Do not use gateways or terminal servers as NIS servers.
- Distribute servers appropriately among client networks. Because an NIS client can bind only to a server on its own subnet, each subnet must have enough servers to accommodate the clients on that subnet.

To Draw an NIS Network Map

It is a very good idea to draw a map of your NIS network, to help with maintenance and troubleshooting in the future. Figure 4-3 shows an example of an NIS network map.

Figure 4-3 Example NIS Network Map



Configuring and Administering an NIS Master Server

An NIS master server holds the source files for all the NIS maps in the domain. Any changes to the NIS maps must be made on the NIS master server. The NIS master server delivers information to NIS clients and supplies the NIS slave servers with up-to-date maps.

An NIS master server must also be an NIS client.

This section explains how to perform the following tasks. Only the first five tasks are required to get your NIS master server up and running.

- To Create the Master passwd File
- To Create the Master group File
- To Create the Master hosts File
- To Enable NIS Master Server Capability
- To Verify Your NIS Master Server Configuration
- To Configure the NIS Master Server to Use a Private passwd File
- To Restrict Client and Slave Server Access to the Master Server
- To Check the Contents of an NIS Map
- To Modify an NIS Map
- To Add an Automounter Map to Your NIS Domain
- To Remove an Automounter Map from Your NIS Domain
- To Add a Slave Server to Your NIS Domain
- To Remove a Slave Server from Your NIS Domain
- To Query BIND for Host Information After Querying NIS
- To Use NIS With Short File Names
- To Configure an HP-UX Master Server in a Domain with Sun Systems

To Create the Master passwd File

1. Copy the /etc/passwd file from each host in your NIS domain to the /etc directory on the host that will be the master server. Name each copy /etc/passwd.*hostname*, where *hostname* is the name of the host it came from.
2. Concatenate all the passwd files together, including the master server's passwd file, into a temporary passwd file, as follows:

   ```
   cd /etc
   cat passwd passwd.hostname1 passwd.hostname2... > passwd.temp
   ```

3. Issue the following command to sort the temporary passwd file by user name:

   ```
   sort -o /etc/passwd.temp -t: -k1,1 /etc/passwd.temp
   ```

4. Examine /etc/passwd.temp for duplicate user names. If you find multiple entries for the same user, edit the file to remove redundant ones. Make sure each user in your network has a unique user name.
5. Issue the following command to sort the temporary passwd file by user ID:

   ```
   sort -o /etc/passwd.temp -t: -k3n,3 /etc/passwd.temp
   ```

6. Examine /etc/passwd.temp for duplicate user IDs. If you find multiple entries with the same user ID, edit the file to change the user IDs so that no two users have the same user ID.
7. Move /etc/passwd.temp (the sorted, edited file) to /etc/passwd. This file will be used to generate the passwd map for your NIS domain.
8. Remove all the /etc/passwd.*hostname* files from the master server.

NOTE NIS does not require that the passwd file be sorted in any particular way. Sorting the passwd file simply makes it easier to find duplicate entries.

For more information, type man 4 passwd or man 1 sort at the HP-UX prompt.

To Create the Master group File

1. Copy the /etc/group file from each host in your NIS domain to the /etc directory on the host that will be the master server. Name each copy /etc/group.*hostname*, where *hostname* is the name of the host it came from.
2. Concatenate all the group files together, including the master server's group file, into a temporary group file, as follows:

   ```
   cd /etc
   cat group group.hostname1 group.hostname2... > group.temp
   ```

3. Issue the following command to sort the temporary group file by group name:

   ```
   sort -o /etc/group.temp -t: -k1,1 /etc/group.temp
   ```

4. Examine /etc/group.temp for duplicate group names. If a group name appears more than once, merge the groups with the same name into one group and remove the duplicate entries.
5. Issue the following command to sort the temporary group file by group ID:

   ```
   sort -o /etc/group.temp -t: -k3n,3 /etc/group.temp
   ```

6. Examine /etc/group.temp for duplicate group IDs. If you find multiple entries with the same group ID, edit the file to change the group IDs so that no two groups have the same group ID.
7. Move /etc/group.temp (the sorted, edited file) to /etc/group. This file will be used to generate the group map for your NIS domain.
8. Remove the /etc/group.*hostname* files from the master server.

NOTE NIS does not require that the group file be sorted in any particular way. Sorting the group file simply makes it easier to find duplicate entries.

For more information, type man 4 group or man 1 sort at the HP-UX prompt.

To Create the Master hosts File

1. Copy the /etc/hosts file from each host in your NIS domain to the /etc directory on the host that will be the master server. Name each copy /etc/hosts.*hostname*, where *hostname* is the name of the host it came from.
2. Concatenate all the hosts files together, including the master server's hosts file, into a temporary hosts file, as follows:

   ```
   cd /etc
   cat hosts hosts.hostname1 hosts.hostname2... > hosts.temp
   ```

3. Issue the following command to sort the temporary hosts file so that duplicate IP addresses are on adjacent lines:

   ```
   sort -o /etc/hosts.temp /etc/hosts.temp
   ```

4. Examine /etc/hosts.temp for duplicate IP addresses. If the same IP address appears in multiple entries, remove all the entries but one. If you need to map an IP address to multiple host names, include them as aliases in a single entry.
5. Issue the following command to sort the temporary hosts file by host name:

   ```
   sort -o /etc/hosts.temp -b -k2,2 /etc/hosts.temp
   ```

6. Examine /etc/hosts.temp for duplicate host names. A host name may be mapped to multiple IP addresses *only* if the IP addresses belong to different LAN cards on the same host. If a host name appears in multiple entries, mapped to IP addresses on different hosts, remove all the entries but one.
7. Examine /etc/hosts.temp for duplicate aliases. No alias should appear in more than one entry.
8. Move /etc/hosts.temp (the sorted, edited file) to /etc/hosts. This file will be used to generate the hosts map for your NIS domain.
9. Remove the /etc/hosts.*hostname* files from the master server.

NOTE NIS does not require that the hosts file be sorted in any particular way. Sorting the hosts file simply makes it easier to find duplicate entries.

For more information, type man 4 hosts or man 1 sort at the HP-UX prompt.

To Enable NIS Master Server Capability

1. Log in as root to the host that will be the master server.
2. On the host that will be the master server, ensure that the `$PATH` environment variable includes the following directory paths:
   - `/var/yp`
   - `/usr/lib/netsvc/yp`
   - `/usr/ccs/bin`
3. Issue the following command to set the NIS domain name:

   ```
   /usr/bin/domainname domainname
   ```

   If your host uses short file names, make sure the first 14 characters of *domainname* uniquely identify your domain among the other NIS domains in your network.
4. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_DOMAIN` variable to the domain name:

   ```
   NIS_DOMAIN=domainname
   ```
5. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_MASTER_SERVER` and `NIS_CLIENT` variables to 1, as follows:

   ```
   NIS_MASTER_SERVER=1
   NIS_CLIENT=1
   ```

   If the host that will be the master server is already a slave server for another domain, set the `NIS_MASTER_SERVER` variable to 1 and the `NIS_SLAVE_SERVER` variable to 0.
6. Issue the following command to create the NIS maps for the domain:

   ```
   /usr/sbin/ypinit -m
   ```

   The `ypinit` script will prompt you for the names of your slave servers. Enter the names of your slave servers in response to the prompt.
7. Issue the following commands to run the NIS startup scripts:

   ```
   /sbin/init.d/nis.server start
   /sbin/init.d/nis.client start
   ```

The master server is now running as both an NIS master server and an NIS client. Next, you must configure the slave servers you listed when you ran the `ypinit` script. See “Configuring and Administering an NIS Slave Server” on page 129.

For more information, see the following man pages: `domainname`(1), `ypinit`(1M), and `ypfiles`(4).

To Verify Your NIS Master Server Configuration

- Log into the master server and issue the following command:

```
/usr/bin/ypwhich -m
```

The `ypwhich -m` command lists all the NIS maps available to the local client and gives the name of the master server that serves each map. In this case, the local host is both the client and the master server. Your display should look something like this, where *mastername* is the name of your local host:

```
# /usr/bin/ypwhich -m
vhe_list mastername
servi.bynp mastername
services.byname mastername
rpc.byname mastername
protocols.bynumber mastername
protocols.byname mastername
rpc.bynumber mastername
passwd.byuid mastername
passwd.byname mastername
networks.byname mastername
networks.byaddr mastername
netgroup.byuser mastername
netgroup.byhost mastername
netgroup mastername
hosts.byname mastername
hosts.byaddr mastername
group.byname mastername
group.bygid mastername
publickey.byname mastername
netid.byname mastername
mail.byaddr mastername
mail.aliases mastername
auto_master mastername
ypservers mastername
```

If you do not see a similar display, see “Troubleshooting NFS Services” on page 173. Type `man 1 ypwhich` for more information on the `ypwhich` command.

To Configure the NIS Master Server to Use a Private passwd File

CAUTION Do not use this procedure if your NIS master server is also a mail server. If the NIS master server uses only a subset of the information in the NIS passwd map, it cannot resolve mail addresses, and mail messages will fail.

1. Log in as root to the NIS master server.
2. Copy the /etc/passwd file to /etc/passwd.yp.
3. Using a text editor, remove users from the /etc/passwd file who should not be allowed access to the NIS master server. Do not include a plus sign (+) in this file.
4. Use a text editor to edit the /var/yp/Makefile file. Change the following line

   ```
   PWFILE=$(DIR)/passwd
   ```

   to the following:

   ```
   PWFILE=$(DIR)/passwd.yp
   ```

5. In the /etc/rc.config.d/namesvrs file, modify the YPPASSWDD_OPTIONS variable. Change the following line

   ```
   YPPASSWDD_OPTIONS="/etc/passwd -m passwd PWFILE=/etc/passwd"
   ```

   to the following:

```
YPPASSWDD_OPTIONS="/etc/passwd.yp -m passwd PWFILE=/etc/passwd.yp"
```

6. Issue the following commands to regenerate the NIS passwd maps from /etc/passwd.yp:

   ```
   cd /var/yp
   /usr/ccs/bin/make passwd
   ```

   This command generates both the passwd.byname and the passwd.byuid maps and pushes them to the slave servers.

   If your slave servers are not up and running yet, run make with the NOPUSH flag set to 1:

   ```
   cd /var/yp
   /usr/ccs/bin/make NOPUSH=1 passwd
   ```

This procedure creates a restricted `/etc/passwd` file that is used only by the NIS master server. The unrestricted `/etc/passwd.yp` file is used to generate the NIS `passwd` maps, which are used by the rest of the hosts in the NIS domain.

For more information, see the following man pages: `passwd`(4), `make`(1), `ypmake`(1M), and `ypinit`(1M).

To Restrict Client and Slave Server Access to the Master Server

1. On the NIS master server, create a file called `/var/yp/securenets`, if it does not already exist.
2. Add lines to the file with the following syntax:

   address_mask IP_address

   The *IP_address* is the internet address of an NIS client, NIS slave server, or subnet that may request NIS information or transfer NIS maps from the NIS master server.

   The *address_mask* indicates which bits in the *IP_address* field are important. If a bit is set in the *address_mask* field, the corresponding bit in the source address of any incoming NIS requests must match the same bit in the *IP_address* field.
3. Issue the following commands to kill and restart the `ypserv` process:

   ```
   /sbin/init.d/nis.server stop
   /sbin/init.d/nis.server start
   ```

If a client or slave host has multiple network interface cards, add a line to the `securenets` file for the IP address of each card.

Type `man 4 securenets` at the HP-UX prompt for more information.

Examples from /var/yp/securenets

The following line from a `/var/yp/securenets` file allows only the NIS client at IP address 10.11.12.13 to request information from the NIS master server. Because every bit is set in the address mask, the source IP address on the NIS request must match exactly, or the master server will not return the requested information.

```
255.255.255.255    10.11.12.13
```

The following line from a `/var/yp/securenets` file allows any host on the network 10.11.12.0 to request NIS information or transfer NIS maps from the master server. The last 8 bits of the IP address are ignored, because the last 8 bits of the address mask are set to 0. Any host whose IP address begins 10.11.12 will be allowed access to the master server.

```
255.255.255.0    10.11.12.13
```

To Check the Contents of an NIS Map

- Issue the following command to verify that an NIS map contains the data you expect it to contain:

  `/usr/bin/ypcat -k` *`mapname`*

The `-k` option lists the key for each item in the map as well as the data associated with the key. For example, in the `netgroup` map, the netgroup name is the key. Without the `-k` option, `ypcat` would list all the data associated with each netgroup name, but not the netgroup name itself.

For more information on the `ypcat` command, type `man 1 ypcat` at the HP-UX prompt.

To Modify an NIS Map

1. Log in as root to the NIS master server.
2. Make your changes to the source file for the NIS map. For example, if you want to change the NIS `hosts` map, make your changes to the `/etc/hosts` file.
3. Issue the following commands to generate the map and push it to the slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make mapname
   ```

   If your slave servers are not up and running yet, run the `make` command with the `NOPUSH` flag set to 1:

   ```
   cd /var/yp
   /usr/ccs/bin/make NOPUSH=1 mapname
   ```

This procedure works for all NIS maps except the `ypservers` map, which has no source file. For instructions on modifying the `ypservers` map, see “To Add a Slave Server to Your NIS Domain” on page 124 or “To Remove a Slave Server from Your NIS Domain” on page 125.

If you make changes to the `passwd`, `group`, or `hosts` maps, regenerate the `netid.byname` map. The `netid.byname` map is a mapping of users to groups, where each user is followed by a list of all the groups to which the user belongs. The `netid.byname` map is generated from the `/etc/passwd` and `/etc/group` files.

For more information, see the following man pages: `make`(1), `ypmake`(1M), `yppush`(1M), and `ypxfr`(1M).

To Add an Automounter Map to Your NIS Domain

1. Log in as root to the NIS master server.
2. In the `/usr/sbin/ypinit` script, use a text editor to add the automounter map to the `MASTER_MAPS` list, as follows:

```
MASTER_MAPS="group.bygid group.byname \
  hosts.byaddr bosts.byname netgroup netgroup.byhost \
  netgroup.byuser networks.byaddr networks.byname passwd.byname \
  passwd.byuid protocols.byname protocols.bynumber rpc.bynumber \
  services.byname vhe_list publickey.byname netid.byname mail.byaddr \
  mail.aliases auto_master rpc.byname servi.bynp auto_mapname"
```

3. In the `/var/yp/Makefile` file, add the automounter map to the list of maps that begins with `all:`, as follows:

```
all: passwd group hosts networks rpc services protocols \
     netgroup aliases publickey netid vhe_list auto_master \
     auto_mapname
```

4. In the `/var/yp/Makefile` file, copy the statement that begins `$(YPDBDIR)/$(DOM)/auto_master.time` to the space below it. Change all occurrences of `auto_master` to the name of the map you are adding.

```
$ (YPDBDIR)/$(DOM)/auto_master.time: $(DIR)/auto_master
        @(sed -e "s/^[ |   ]*//g" -e "/^#/d" -e s/#.*$$// <
$(DIR)/auto_master $(CHKPIPE)) |
            $(MAKEDBM) - $(YPDBDIR) /$(DOM)/auto_master;
        @touch $(YPDBDIR)/$(DOM)/auto_master.time;
        @echo "updated auto_master";
        @if [ ! $(NOPUSH) ]; then $(YPPUSH) -d $(DOM) auto_master; fi
        @if [ ! $(NOPUSH) ]; then echo "pushed auto_master"; fi

$ (YPDBDIR)/$(DOM)/auto_mapmame.time: $(DIR)/auto_mapname
        @(sed -e "s/^[ |   ]*//g" -e "/^#/d" -e s/#.*$$// <
$(DIR)/auto_mapname $(CHKPIPE)) |
            $(MAKEDBM) - $(YPDBDIR) /$(DOM)/auto_mapname;
        @touch $(YPDBDIR)/$(DOM)/auto_mapname.time;
        @echo "updated auto_mapname";
        @if [ ! $(NOPUSH) ]; then $(YPPUSH) -d $(DOM) auto_mapname; fi
        @if [ ! $(NOPUSH) ]; then echo "pushed auto_mapname"; fi
```

5. In the `/var/yp/Makefile` file, copy the statement that begins `auto_master:` to the space below it. Change `auto_master` to `auto_`*mapname*, and change both occurrences of `auto_master.time` to `auto_`*mapname*`.time`.

```
auto_master:
    @if [ $(NOPUSH) ]; then $(MAKE) $(MFLAGS) -k \
          $(YPDBDIR)/$(DOM)/auto_master.time DOM=$(DOM) DIR=$(DIR); \
    else $(MAKE) $(MFLAGS) -k $(YPDBDIR)/$(DOM)/auto_master.time \
           DOM=$(DOM) DIR=$(DIR) NOPUSH=$(NOPUSH);fi

auto_mapname:
    @if [ $(NOPUSH) ]; then $(MAKE) $(MFLAGS) -k \
          $(YPDBDIR)/$(DOM)/auto_mapname.time DOM=$(DOM) DIR=$(DIR); \
    else $(MAKE) $(MFLAGS) -k $(YPDBDIR)/$(DOM)/auto_mapname.time \
           DOM=$(DOM) DIR=$(DIR) NOPUSH=$(NOPUSH);fi
```

6. Issue the following commands to generate the map:

   ```
   cd /var/yp
   /usr/ccs/bin/make NOPUSH=1 auto_mapname
   ```

7. If you have slave servers configured in your domain, log into each slave server and issue the following command to copy the new map to the slave server:

   ```
   /usr/sbin/ypxfr auto_mapname
   ```

For more information, see the man page for `ypinit`(1M), `make`(1), `ypmake`(1M), or `ypxfr`(1M).

To Remove an Automounter Map from Your NIS Domain

1. Log in as root to the NIS master server.
2. In the `/usr/sbin/ypinit` script, use a text editor to remove the map name from the `MASTER_MAPS` list.
3. In the `/var/yp/Makefile` file, remove the map from the list of maps that begins with `all:`.
4. In the `/var/yp/Makefile` file, remove the statement that begins `$(YPDBDIR)/$(DOM)/auto_`*mapname*`.time`. For example, if you were removing the `auto_home` map, you would remove the following lines:

```
$ (YPDBDIR)/$(DOM)/auto_home.time: $(DIR)/auto_home
        @(sed -e "s/^[ |   ]*//g" -e "/^#/d" -e s/#.*$$// <
$(DIR)/auto_home $(CHKPIPE)) |
             $(MAKEDBM) - $(YPDBDIR) /$(DOM)/auto_home;
        @touch $(YPDBDIR)/$(DOM)/auto_home.time;
        @echo "updated auto_home";
        @if [ ! $(NOPUSH) ]; then $(YPPUSH) -d $(DOM) auto_home; fi
        @if [ ! $(NOPUSH) ]; then echo "pushed auto_home"; fi
```

5. In the `/var/yp/Makefile` file, remove the statement that begins `auto_`*mapname*`:`. For example, if you were removing the `auto_home` map, you would remove the following lines:

```
auto_home:
    @if [ $(NOPUSH) ]; then $(MAKE) $(MFLAGS) -k \
          $(YPDBDIR)/$(DOM)/auto_home.time DOM=$(DOM) DIR=$(DIR); \
    else $(MAKE) $(MFLAGS) -k $(YPDBDIR)/$(DOM)/auto_home.time \
           DOM=$(DOM) DIR=$(DIR) NOPUSH=$(NOPUSH);fi
```

6. On the master and on each of the slave servers, remove the map files, *mapname*`.dir` and *mapname*`.pag` from the directory where your maps are stored. The directory is called `/var/yp/`*domainname*, where *domainname* is the name of your NIS domain. For example, if you were removing the `auto_home` map from the `Finance` domain, you would issue the following commands on the master server and on each of the slave servers:

   ```
   cd /var/yp/Finance
   rm auto_home.dir auto_home.pag
   ```

For more information, see the man pages `ypinit`(1M), `make`(1), `ypmake`(1M), and `ypfiles`(4).

To Add a Slave Server to Your NIS Domain

1. Log in as root to the NIS master server.
2. Issue the following command, where *domainname* is the name of the domain to which you want to add the slave server:

   ```
   cd /var/yp/domainname
   ```

3. Issue the following command to create an editable ASCII text file from the `ypservers` map:

   ```
   /usr/sbin/makedbm -u ypservers > tempfile
   ```

4. Use a text editor to add the name of the new server to the ASCII file, `tempfile`.
5. Issue the following command to regenerate the `ypservers` map from the ASCII file:

   ```
   /usr/sbin/makedbm tempfile ypservers
   ```

6. Log in as root to the new slave server and configure it as an NIS slave server. See “Configuring and Administering an NIS Slave Server” on page 129.

For more information, see the man page for `makedbm`(1M) or `ypfiles`(4).

To Remove a Slave Server from Your NIS Domain

1. Log in as root to the NIS master server.
2. Issue the following commands to create an editable ASCII text file from the `ypservers` map:

   ```
   cd /var/yp/domainname
   /usr/sbin/makedbm -u ypservers > tempfile
   ```

3. Use a text editor to remove the name of the slave server from the ASCII file, `tempfile`.
4. Issue the following command to regenerate the `ypservers` map from the ASCII file:

   ```
   /usr/sbin/makedbm tempfile ypservers
   ```

5. Log in as root to the slave server.
6. Remove all the map files from the map directory, and remove the map directory. The directory is called `/var/yp/`*domainname*, where *domainname* is the name of your NIS domain. For example, if you were removing a slave server from the `Finance` domain, you would issue the following commands:

   ```
   cd /var/yp/Finance
   rm *
   cd ..
   rmdir Finance
   ```

7. If the slave is not a slave server in any other NIS domain, use a text editor to set the `NIS_SLAVE_SERVER` variable to 0 in the `/etc/rc.config.d/namesvrs` file.

   ```
   NIS_SLAVE_SERVER=0
   ```

8. If the slave is not a server in any other NIS domain, issue the following command to turn off NIS server capability:

   ```
   /sbin/init.d/nis.server stop
   ```

For more information, see the man pages `makedbm`(1M) and `ypfiles`(4).

To Query BIND for Host Information After Querying NIS

This section tells you how to set up **server-side hostname fallback**, which causes your NIS servers to query BIND for host information after querying NIS. A server will search the NIS `hosts` database first, but if the `hosts` database does not contain the requested information, the server will query the BIND name service. The server will return the host information to the clients through NIS.

1. Configure your NIS servers as BIND name servers, or install an `/etc/resolve.conf` file on each server that allows it to query a BIND name server. See *Installing and Administering Internet Services* for more information.
2. On the NIS master server, in the `/var/yp/Makefile` file, set the `B` variable to `-b`, as follows:

   ```
   B=-b
   ```

3. Issue the following command on the master server to change the modification time on `/etc/hosts` so that `make` will regenerate the `hosts` database:

   ```
   /usr/bin/touch /etc/hosts
   ```

4. Issue the following commands to regenerate the NIS maps on the master server and push them to the NIS slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make
   ```

5. On all the NIS servers in your domain, change the `hosts` line in the `/etc/nsswitch.conf` file to the following:

   ```
   hosts: nis dns files
   ```

Hewlett-Packard recommends that you use the Name Service Switch on your NIS clients instead of server-side hostname fallback. However, if your NIS clients are PCs that do not have a feature like the Name Service Switch, use the server-side hostname fallback described in this section if you want to force BIND lookups after NIS lookups. See “Configuring the Name Service Switch” on page 153.

To Use NIS With Short File Names

1. Make sure the first 14 characters of your domain name uniquely identify your domain among the other NIS domains in your network.
2. If you plan to use NIS to manage your automounter maps, keep the automounter map names to 10 characters or fewer.
3. Log in as root to the NIS master server.
4. In the `/var/yp/Makefile` file, uncomment all the lines between `START OF EXAMPLE` and `END OF EXAMPLE`. (Remove the sharp sign [#] from the beginning of each line.) Do not uncomment the `START OF EXAMPLE` and `END OF EXAMPLE` lines.
5. In the `/var/yp/Makefile` file, delete everything after the `END OF EXAMPLE` line.

This procedure causes your NIS master server to use HP's proprietary `ypmake` script instead of the `Makefile`. The `Makefile` does not support short filenames, but `ypmake` does. Type `man 1M ypmake` at the HP-UX prompt for more information.

To Configure an HP-UX Master Server in a Domain with Sun Systems

1. Log in as root to the host that will be the master server.
2. If you have customized your HP Makefile, move it to `/var/yp/Makefile.hp`.
3. Copy your Sun Makefile into the `/var/yp` directory on the HP system.

   If your Sun Makefile is not called `Makefile`, use a text editor to set the `MAKEFILE_NAME` variable to the name of your Sun Makefile in the `/usr/sbin/ypinit` script.
4. If you have customized your HP Makefile, add those changes into your Sun Makefile.
5. In the `/usr/sbin/ypinit` script on the HP host that will be the master server, add the `netmasks.byaddr`, `bootparams`, `ethers.byaddr`, and `ethers.byname` maps to the `MASTER_MAPS` variable.
6. On one of your Sun systems, locate or create an `/etc/ethers` file, an `/etc/bootparams` file, and an `/etc/netmasks` file that contain all the information required by the Sun systems in your NIS domain.
7. Copy the `/etc/ethers`, `/etc/bootparams`, and `/etc/netmasks` files to the HP host that will be the master server.
8. Follow the instructions in “To Enable NIS Master Server Capability” on page 113.

Configuring and Administering an NIS Slave Server

An NIS slave server provides information to NIS clients, taking some load off the NIS master server and substituting for the master server when it is down. The NIS maps are created on the NIS master server and then transferred from the master server to the slave servers. Changes to NIS maps must be made on the NIS master server, which then pushes the changed maps to the NIS slave servers.

An NIS slave server must also be an NIS client.

The NIS master server must be configured and running before you start your slave servers.

This section explains how to perform the following tasks:

- To Edit the Slave Server's passwd File
- To Edit the Slave Server's group File
- To Enable NIS Slave Server Capability
- To Verify Your NIS Slave Server Configuration
- To Schedule Regular Map Transfers from the NIS Master Server
- To Restrict Access to the Slave Server

To Edit the Slave Server's passwd File

- Remove all users from the /etc/passwd file except the root user and the system entries required for your system to boot. By convention, system entries usually have user IDs less than 100, so you can remove all entries with user IDs of 100 or greater.
- Add the following entry as the last line in the /etc/passwd file:

  ```
  +::-2:60001:::
  ```

  The plus sign (+) causes processes to consult NIS for any user information not found in the local /etc/passwd file.

  The -2 in the user ID field restricts the access of people who may attempt to log in using "+" as a valid user name when NIS is not running. Anyone who successfully logs in as "+" will be granted only the access permissions of user nobody.

CAUTION Do not put an asterisk (*) in the password field on HP systems. On Sun systems, an asterisk in the password field prevents people from logging in as "+" when NIS is not running. However, on HP systems, the asterisk prevents all users from logging in when NIS is running.

The changes you make to the /etc/passwd file on an NIS slave server are the same changes you make on an NIS client. Following is an example /etc/passwd file on an NIS slave server:

```
root:0AnhFBmriKvHA:0:3::/:/bin/ksh
daemon:*:1:5::/:/bin/sh
bin:*:2:2::/bin:/bin/sh
adm:*:4:4::/usr/adm:/bin/sh
uucp:*:5:3::/usr/spool/uucppublic:/usr/lib/uucp/uucico
lp:*:9:7::/usr/spool/lp:/bin/sh
hpdb:*:27:1:ALLBASE:/:/bin/sh
+::-2:60001:::
```

For more information, type man 4 passwd at the HP-UX prompt.

To Edit the Slave Server's group File

- Remove all groups from the /etc/group file except the group entries required for your system to boot.
- Add the following entry as the last line in the /etc/group file:

  ```
  +:*:*
  ```

  The plus sign (+) causes processes to consult NIS for any group information not found in the local /etc/group file. The asterisk (*) in the password field prevents people from using the plus sign as a valid group name if NIS is not running.

The changes you make to the /etc/group file on an NIS slave server are the same changes you make on an NIS client. Following is an example /etc/group file on an NIS slave server:

```
root::0:rootl,sam
other::1:
bin::2:
sys::3:
adm::4:
daemon::5:
mail::6:
lp::7:
+:*:*
```

For more information, type man 4 group at the HP-UX prompt.

To Enable NIS Slave Server Capability

1. Make sure the NIS master server is already configured and running NIS.
2. Log in as root to the host that will be the slave server.
3. On the host that will be the slave server, ensure that the `$PATH` environment variable includes the following directory paths:
   - `/var/yp`
   - `/usr/lib/netsvc/yp`
   - `/usr/ccs/bin`
4. Issue the following command to set the NIS domain name:

   `/usr/bin/domainname` *domainname*

   where *domainname* is the same as the domain name on the NIS master server.
5. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_DOMAIN` variable to the domain name:

   `NIS_DOMAIN=`*domainname*
6. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_SLAVE_SERVER` and `NIS_CLIENT` variables to 1, as follows:

   ```
   NIS_SLAVE_SERVER=1
   NIS_CLIENT=1
   ```

   If the slave server is a master server in another NIS domain, set the `NIS_MASTER_SERVER` variable to 1 and the `NIS_SLAVE_SERVER` variable to 0. The `yppasswdd` daemon, which is required on the master server, is started only if `NIS_MASTER_SERVER=1`.
7. Issue the following command to set up the NIS slave server and copy the NIS maps from the master server:

   `/usr/sbin/ypinit -s` *NIS_server_name* `[DOM=`*domainname*`]`

   The *NIS_server_name* is the name of the master server or a slave server that has a complete set of up-to-date maps for the domain. If the slave server will serve a domain different from the one set by the `domainname` command, specify the *domainname* after the *NIS_server_name*.
8. Issue the following commands to run the NIS startup scripts:

```
/sbin/init.d/nis.server start
/sbin/init.d/nis.client start
```

In order to receive map updates from the NIS master server, you must add the new slave server to the `ypservers` map on the master server. See “To Add a Slave Server to Your NIS Domain” on page 124.

For more information, see the following man pages: `domainname`(1), `ypinit`(1M), and `ypfiles`(4).

To Verify Your NIS Slave Server Configuration

1. Log in as root to the slave server.
2. In the `/etc/rc.config.d/namesvrs` file, add `-ypset` to the `YPBIND_OPTIONS` variable:

   ```
   YPBIND_OPTIONS="-ypset"
   ```

3. Issue the following commands to restart `ypbind` (the NIS client process) on the slave server:

   ```
   /sbin/init.d/nis.client stop
   /sbin/init.d/nis.client start
   ```

4. Issue the following command to force the NIS client process on the slave server to bind to the server process on the same host:

   `/usr/sbin/ypset` *slave_server_name*

5. Issue the following command to check whether the NIS slave server is working:

   ```
   /usr/bin/ypwhich
   ```

   The `ypwhich` command should return the host name of the slave server. If the `ypwhich` command does not return the name of the slave server, see "Troubleshooting NFS Services" on page 173.

6. In the `/etc/rc.config.d/namesvrs` file, remove `-ypset` from the `YPBIND_OPTIONS` variable:

   ```
   YPBIND_OPTIONS=""
   ```

7. Issue the following commands to restart `ypbind` (the NIS client process) on the slave server:

   ```
   /sbin/init.d/nis.client stop
   /sbin/init.d/nis.client start
   ```

For more information, see the following man pages: `ypbind`(1M), `ypset`(1M), and `ypwhich`(1).

To Schedule Regular Map Transfers from the NIS Master Server

1. Log in as root to the slave server.
2. Copy the `ypxfr_1perday`, `ypxfr_2perday`, and `ypxfr_1perhour` scripts from the `/usr/newconfig/var/yp` directory to the `/var/yp` directory:

   ```
   cp /usr/newconfig/var/yp/ypxfr_1perday /var/yp
   cp /usr/newconfig/var/yp/ypxfr_2perday /var/yp
   cp /usr/newconfig/var/yp/ypxfr_1perhour /var/yp
   ```

3. Create a `crontab` file that invokes these files at regular times. Following is an example `crontab` file:

   ```
   0 21 * * * /var/yp/ypxfr_1perday
   30 5,19 * * * /var/yp/ypxfr_2perday
   15 * * * * /var/yp/ypxfr_1perhour
   ```

   This file runs the `ypxfr_1perday` script at 9:00 PM every night. It runs the `ypxfr_2perday` script at 5:30 AM and 7:30 PM every day. It runs the `ypxfr_1perhour` at 15 minutes past every hour.

4. Issue the following command to enter the file into `crontab`,

   `crontab` *`filename`*

   where *`filename`* is the `crontab` file you just created.

If you have created customized NIS maps for your domain, you will have to add them to the appropriate scripts. You can also use the scripts provided as templates for creating your own scripts.

In some domains, transferring the `passwd` maps once per hour generates too much network traffic. If you find this is the case, schedule transfers of the `passwd` maps for less frequent intervals.

If you have multiple slave servers, schedule map transfers for different times on different servers, so all the servers are not performing transfers at the same time.

For more information, see the following man pages: `cron`(1M), `crontab`(1), and `ypxfr`(1M).

To Restrict Access to the Slave Server

1. On the NIS slave server, create a file called `/var/yp/securenets`, if it does not already exist.
2. Add lines to the file with the following syntax:

   address_mask IP_address

   The *IP_address* is the internet address of an NIS client, NIS slave server, or subnet that may request NIS information or transfer NIS maps from the NIS master server.

   The *address_mask* indicates which bits in the *IP_address* field are important. If a bit is set in the *address_mask* field, the corresponding bit in the source address of any incoming NIS requests must match the same bit in the *IP_address* field.
3. Issue the following commands to kill and restart the `ypserv` process:

```
/sbin/init.d/nis.server stop
/sbin/init.d/nis.server start
```

If a client or slave host has multiple network interface cards, add a line to the `securenets` file for the IP address of each card.

Type `man 4 securenets` at the HP-UX prompt for more information.

Examples from /var/yp/securenets

The following line from a `/var/yp/securenets` file allows only the NIS client at IP address 10.11.12.13 to request information from the NIS slave server. Because every bit is set in the address mask, the source IP address on the NIS request must match exactly, or the slave server will not return the requested information.

```
255.255.255.255    10.11.12.13
```

The following line from a `/var/yp/securenets` file allows any host on the network 10.11.12.0 to request NIS information or transfer NIS maps from the slave server. The last 8 bits of the IP address are ignored, because the last 8 bits of the address mask are set to 0. Any host whose IP address begins 10.11.12 will be allowed access to the slave server.

```
255.255.255.0    10.11.12.13
```

Configuring and Administering an NIS Client

An NIS client gets its configuration information from an NIS master server or an NIS slave server. When an NIS client is started, it sends out a broadcast message requesting a server. Any server on the client's network that holds the NIS maps for the client's domain may respond to the message. The NIS client "binds" to the first server to respond to its broadcast message, and that server answers all the client's queries for information.

This section explains how to perform the following tasks. Only the first five tasks are necessary for getting your NIS client up and running.

- To Edit the NIS Client's passwd File
- To Edit the NIS Client's group File
- To Enable NIS Client Capability
- To Verify Your NIS Client Configuration
- To Tell Users How to Use yppasswd
- To Prevent a Client from Binding to Unknown Servers
- To Bind an NIS Client to a Server on a Different Subnet

To Edit the NIS Client's passwd File

- Remove all users from the /etc/passwd file except the root user and the system entries required for your system to boot. By convention, system entries usually have user IDs less than 100, so you can remove all entries with user IDs of 100 or greater.
- Add the following entry as the last line in the /etc/passwd file:

  ```
  +::-2:60001:::
  ```

  The plus sign (+) causes processes to consult NIS for any user information not found in the local /etc/passwd file.

  The -2 in the user ID field restricts the access of people who may attempt to log in using "+" as a valid user name when NIS is not running. Anyone who successfully logs in as "+" will be granted only the access permissions of user nobody.

CAUTION Do not put an asterisk (*) in the password field on HP systems. On Sun systems, an asterisk in the password field prevents people from logging in as "+" when NIS is not running. However, on HP systems, the asterisk prevents all users from logging in when NIS is running.

The changes you make to the /etc/passwd file on an NIS client are the same changes you make on an NIS slave server. Following is an example /etc/passwd file on an NIS client:

```
root:0AnhFBmriKvHA:0:3: :/:/bin/ksh
daemon:*:1:5::/:/bin/sh
bin:*:2:2::/bin:/bin/sh
adm:*:4:4::/usr/adm:/bin/sh
uucp:*:5:3::/usr/spool/uucppublic:/usr/lib/uucp/uucico
lp:*:9:7::/usr/spool/lp:/bin/sh
hpdb:*:27:1:ALLBASE:/:/bin/sh
+::-2:60001:::
```

For more information, type man 4 passwd at the HP-UX prompt.

To Edit the NIS Client's group File

- Remove all groups from the `/etc/group` file except the group entries required for your system to boot.
- Add the following entry as the last line in the `/etc/group` file:

  ```
  +:*:*
  ```

  The plus sign (+) causes processes to consult NIS for any group information not found in the local `/etc/group` file. The asterisk (*) in the password field prevents people from using the plus sign as a valid group name if NIS is not running.

The changes you make to the `/etc/group` file on an NIS client are the same changes you make on an NIS slave server. Following is an example `/etc/group` file on an NIS client:

```
root::0:rootl,sam
other::1:
bin::2:
sys::3:
adm::4:
daemon::5:
mail::6:
lp::7:
+:*:*
```

For more information, type `man 4 group` at the HP-UX prompt.

To Enable NIS Client Capability

1. Make sure at least one NIS master or slave server is running on the client's subnetwork.
2. Log in as root to the NIS client.
3. On the NIS client, ensure that the $PATH environment variable includes the following directory paths:
   - `/var/yp`
   - `/usr/lib/netsvc/yp`
   - `/usr/ccs/bin`
4. Issue the following command to set the NIS domain name:

   ```
   /usr/bin/domainname domainname
   ```

   where *domainname* is a domain served by an NIS server on the client's subnetwork.
5. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_DOMAIN` variable to the domain name:

   ```
   NIS_DOMAIN=domainname
   ```
6. In the `/etc/rc.config.d/namesvrs` file, set the `NIS_CLIENT` variable to 1, as follows:

   ```
   NIS_CLIENT=1
   ```
7. Issue the following command to run the NIS startup script:

   ```
   /sbin/init.d/nis.client start
   ```

For more information, see the following man pages: `domainname`(1), `ypbind`(1M), and `nsswitch.conf`(4).

To Verify Your NIS Client Configuration

- Log into the NIS client and issue the following command:

```
/usr/bin/ypwhich -m
```

The `ypwhich -m` command lists all the NIS maps available to the client and gives the name of the master server that serves each map. Your display should look something like this, where *mastername* is the name of the master server for your domain:

```
# /usr/bin/ypwhich -m
vhe_list mastername
servi.bynp mastername
services.byname mastername
rpc.byname mastername
protocols.bynumber mastername
protocols.byname mastername
rpc.bynumber mastername
passwd.byuid mastername
passwd.byname mastername
networks.byname mastername
networks.byaddr mastername
netgroup.byuser mastername
netgroup.byhost mastername
netgroup mastername
hosts.byname mastername
hosts.byaddr mastername
group.byname mastername
group.bygid mastername
publickey.byname mastername
netid.byname mastername
mail.byaddr mastername
mail.aliases mastername
auto_master mastername
ypservers mastername
```

If you do not see a similar display, see “Troubleshooting NFS Services” on page 173. Type `man 1 ypwhich` for more information on the `ypwhich` command.

To Tell Users How to Use yppasswd

- Tell all the users in your NIS domain that they must use `/usr/bin/yppasswd` or `passwd -r nis` instead of the `passwd` command when they want to change their login passwords.
- Tell users that, when they want to change their login passwords, they should do so just before they leave for the day. This will allow time for the updated NIS maps on the master server to be pushed to the slave servers.

The `yppasswd` command is a link to the `passwd -r nis` command. It changes the `/etc/passwd` file on the NIS master server, regenerates the NIS `passwd` maps from the updated `/etc/passwd` file, and then pushes the NIS `passwd` maps to the slave servers.

For more information, see the following man pages: `yppasswd`(1), `yppasswdd`(1M), `passwd`(1), `ypxfr`(1M), and `yppush`(1M).

To Prevent a Client from Binding to Unknown Servers

1. On the NIS client, create a file called `/var/yp/secureservers`, if it does not already exist.
2. Add lines to the file with the following syntax:

   address_mask IP_address

   The *IP_address* is the internet address of an NIS server or the subnet of an NIS server from which the client will accept NIS information.

   The *address_mask* indicates which bits in the *IP_address* field are important. If a bit is set in the *address_mask* field, the corresponding bit in the address of any NIS server must match the same bit in the *IP_address* field.
3. Issue the following commands to kill and restart the `ypbind` process:

   ```
   /sbin/init.d/nis.client stop
   /sbin/init.d/nis.client start
   ```

If an NIS server host has multiple network interface cards, add a line to the `secureservers` file for the IP address of each card.

If you start `ypbind` with the `-ypset` option and issue the `ypset` command to bind to a specific server, the `/var/yp/secureservers` file is ignored, and the client may bind to any server.

Type `man 1M ypbind` at the HP-UX prompt for more information.

Examples from /var/yp/secureservers

The following line from a `/var/yp/secureservers` file allows the NIS client to bind only to the server at IP address 20.21.22.23. Because every bit is set in the address mask, the IP address of the NIS server must match the *IP_address* field exactly, or the client will not bind to it.

```
255.255.255.255     20.21.22.23
```

The following line from a `/var/yp/secureservers` file allows the client to bind to any NIS server on the network 20.21.22.0. The last 8 bits of the server's IP address are ignored, because the last 8 bits of the address mask are set to 0. The client may bind to any server whose IP address begins 20.21.22.

```
255.255.255.0     20.21.22.23
```

To Bind an NIS Client to a Server on a Different Subnet

Hewlett-Packard recommends that you configure a server on each subnet where you have NIS clients; however, if you cannot do that, follow these steps to force an NIS client to bind to a server on a different subnet:

1. Log in as root to the NIS client.
2. Add the `-ypset` option to the `YPBIND_OPTIONS` variable in the `/etc/rc.config.d/namesvrs` file, as follows:

   ```
   YPBIND_OPTIONS="-ypset"
   ```

3. In the `/etc/rc.config.d/namesvrs` file, set the `YPSET_ADDR` variable to the IP address of an NIS server, as in the following example:

   ```
   YPSET_ADDR="15.13.115.168"
   ```

4. Issue the following commands to restart the NIS client:

   ```
   /sbin/init.d/nis.client stop
   /sbin/init.d/nis.client start
   ```

If the server you specify in the `ypset` command is unavailable when your client boots up, your client will broadcast a request for a server to its local network. If no server exists on the local network, the client will hang.

For more information, type `man 1M ypset` or `man 1M ypbind`.

Configuring and Administering Secure RPC

Configuring secure RPC allows you to write applications that use secure RPC. You must be running NIS in order to use secure RPC.

NOTE Secure NFS, the ability to export and mount directories with the `secure` option, is not supported on HP-UX.

Configuring and administering secure RPC involves the following tasks:

- To Have Users Create their Secure RPC Keys

  or

  To Create Secure RPC Keys for Users
- To Create Secure RPC Keys for Hosts
- To Tell Users How to Use Secure RPC

To Have Users Create their Secure RPC Keys

1. In the `/etc/publickey` file on the NIS master server, make sure the entry for user `nobody` exists and is not commented out (is not preceded by #).
2. Tell each user in your NIS domain to issue the `chkey` command:

   ```
   /usr/bin/chkey
   ```

   At the `Password` prompt, the user should enter his or her login password.

The `chkey` command displays a message saying it is generating a key for `unix.`*UID*`@`*NIS_domain*. This string identifies the user in the `publickey.byname` NIS map. *UID* is the user ID of the user for whom the key is being generated, and *NIS_domain* is the default NIS domain, returned by the `domainname` command.

The secure RPC key is encrypted with the user's login password. The `/usr/bin/yppasswd` command reencrypts the secure RPC key with the new password whenever a user changes the login password.

In order for users to create keys for themselves with the `chkey` command, the `publickey.byname` map must have an entry for user `nobody`. If you remove the entry for user `nobody`, users can change their secure RPC keys with the `chkey` command, but they cannot create keys if they do not already have them.

For more information, see the following man pages: `publickey`(4), `chkey`(1), and `yppasswd`(1).

To Create Secure RPC Keys for Users

Use this procedure if you do not want users to be able to create their own secure RPC keys.

1. Log in as root to the NIS master server.
2. Comment out the entry in the `/etc/publickey` file for user `nobody`. (Insert a sharp sign [#] as the first character on the line.)
3. Issue the following commands to regenerate the `publickey.byname` map from the `/etc/publickey` file and push it to the slave servers:

   ```
   cd /var/yp
   /usr/ccs/bin/make publickey
   ```

4. Issue the `newkey -u` command for each user in your NIS domain:

   ```
   # /usr/sbin/newkey -u username
   ```

   Enter a password when prompted for it by the `newkey -u` command.
5. Tell users the passwords you assigned for them. Users should issue the `/usr/bin/keylogin` command, using the passwords you assigned. Then, they should issue the `/usr/bin/yppasswd` command to change their login passwords. The `yppasswd` command will reencrypt their secure RPC keys with their new login passwords.

The `newkey -u` command displays a message saying it is adding a key for `unix.`*UID@NIS_domain*. This string identifies the user in the `publickey.byname` NIS map. *UID* is the user ID of the user for whom the key is being generated, and *NIS_domain* is the default NIS domain, returned by the `domainname` command.

For more information, see the following man pages: `publickey`(4), `newkey`(1M), `chkey`(1), `keylogin`(1), `yppasswd`(1), `make`(1), `ypmake`(1M), and `yppush`(1M).

To Create Secure RPC Keys for Hosts

1. Log in as root to the NIS master server.
2. Issue the `newkey -h` command for each host in your NIS domain:

   ```
   # /usr/sbin/newkey -h hostname
   ```

3. Enter the root password for *hostname* when prompted for it by the `newkey -h` command.
4. On each host for which you have just created a secure RPC key, log in as root. This registers the secure RPC password with the `/usr/sbin/keyserv` daemon.

The `newkey -h` command displays a message saying it is adding a key for `unix.`*hostname*`@`*NIS_domain*. This string identifies the host in the `publickey.byname` NIS map.

Whenever you change the root password with the `passwd` command, the `passwd` command automatically reencrypts the secure RPC key with the new root password.

For more information, see the following man pages: `newkey`(1M), `publickey`(4), `passwd`(1), and `keyserv`(1M).

To Tell Users How to Use Secure RPC

Tell the users who require secure RPC authorization to follow these guidelines:

- If you allow users to create their own secure RPC keys with the `chkey` command, they should enter their login passwords at the `Password` prompt.
- If you use the `newkey -u` command to add users to the `publickey` database, users should issue the `/usr/bin/keylogin` command using the password you assigned. Then, they should issue the `/usr/bin/yppasswd` command to change their login passwords. The `yppasswd` command will automatically reencrypt their secure RPC keys with their new passwords.
- When users log into a host without supplying a password (for example, when they use `rlogin` to log into a host that has their local host configured in `/etc/hosts.equiv`), they should issue the `/usr/bin/keylogin` command after logging in, to register the secure RPC password with the `/usr/sbin/keyserv` daemon.

For more information, see the following man pages: `publickey`(4), `newkey`(1M), `chkey`(1), `keylogin`(1), `yppasswd`(1), `rlogin`(1).

Summary of NIS Commands

Table 4-1 Summary of NIS Commands

chkey(1)	Creates or changes a secure RPC key.
domainname(1)	Sets or displays the name of the NIS domain.
keylogin(1)	Decrypts and stores a secure RPC key. keylogin is called when a user logs in, but the user must issue keylogin if no password was provided at login or if a password other than the login password was used to encrypt the secure RPC key.
keylogout(1)	Deletes a stored decrypted secure RPC key.
makedbm(1M)	Generates an NIS map from an ASCII input file.
newkey(1M)	Creates a secure RPC key for a user or host.
ypcat(1)	Prints all the values in an NIS map.
ypinit(1M)	Sets up an NIS master server or slave server.
ypmake(1M)	Generates one or more NIS maps from ASCII files and optionally pushes them to NIS slave servers. /var/yp/Makefile and make(1) do the same thing.
ypmatch(1)	Prints the values associated with one or more selected keys in an NIS map.
yppasswd(1)	Changes a login password stored in the NIS passwd map.
yppoll(1M)	Returns the name of the master server for an NIS map and the time when the map was built.
yppush(1M)	Forces NIS slave servers to transfer one or more NIS maps from the master server. Slave servers use ypxfr to transfer the maps. ypmake calls yppush unless it is invoked with NOPUSH=1.
ypset(1M)	Tells an NIS client process (ypbind[1M]) to bind to a specified NIS server. ypset can be used only if ypbind is invoked with the -ypset option.

`ypwhich`(1)	Returns the name of the NIS server for the local client or the name of the NIS master server for one or more NIS maps.
`ypxfr`(1M)	Transfers one or more NIS maps from a master server to the local slave server. A slave server calls `ypxfr` when `yppush` is executed on the master server.

5 Configuring the Name Service Switch

The Name Service Switch determines where your host will look for the information that is traditionally stored in the following files:

- automounter maps (like `/etc/auto_master` and `/etc/auto_home`)
- `/etc/hosts`
- `/etc/netgroup`
- `/etc/networks`
- `/etc/protocols`
- `/etc/rpc`
- `/etc/services`

You can configure your host to look for each type of information in NIS or the local `/etc` file. You can configure your host to consult either or both of these sources, in any order.

For host information (host names and IP addresses), you can configure your host to consult BIND (DNS) in addition to NIS or the local `/etc/hosts` file.

The default Name Service Switch configuration is adequate for most installations, so you probably do not have to change it. The default configuration is explained in “Default Configuration” on page 160.

The ability to consult more than one name service for host information is often called **hostname fallback**. The Name Service Switch provides **client-side hostname fallback**, because it is used by client-side programs (for example, `gethostbyname`), which request host information.

NIS allows you to configure a **server-side hostname fallback**, which causes the NIS server to query BIND when it fails to find requested host information in its database. The NIS server then returns the host information to the client through NIS. This server-side hostname fallback is intended for use with clients like PCs that do not have a feature like the Name Service Switch. Hewlett-Packard recommends that you use the Name Service Switch if possible, instead of the server-side hostname fallback provided by NIS. For more information on the NIS server-side hostname fallback, see “To Query BIND for Host Information After Querying NIS” on page 126.

You can use SAM to configure the Name Service Switch. Type `sam` at the HP-UX prompt.

This chapter tells you how to configure the Name Service Switch. It contains the following sections:

- Customizing the nsswitch.conf File
- Syntax of the nsswitch.conf File
- Default Configuration
- Troubleshooting the Name Service Switch

NOTE Configuring the Name Service Switch is a separate task from configuring the name services themselves. You must also configure the name services before you can use them. The Name Service Switch just determines which name services are queried and in what order.

Customizing the `nsswitch.conf` File

The configuration file for the Name Service Switch is called `/etc/nsswitch.conf`. If this file does not exist, the system has a default Name Service Switch configuration, described in "Default Configuration" on page 160, later in this chapter.

Sample Name Service Switch configurations are located in the `/usr/examples/nsswitch` directory.

Following are some suggestions for customizing your Name Service Switch configuration:

- If you want your system to consult the local `/etc/netgroup` file when it fails to find a netgroup in the NIS `netgroup` database, create or modify the `netgroup` line in the `/etc/nsswitch.conf` file as follows:

  ```
  netgroup: nis [NOTFOUND=continue] files
  ```

- If you want your system to consult BIND (DNS) when it fails to find a host name in NIS, create or modify the `hosts` line in the `/etc/nsswitch.conf` file as follows:

  ```
  hosts: nis [NOTFOUND=continue] dns files
  ```

  With this configuration, if NIS does not contain the requested information, and BIND is not configured, the `/etc/hosts` file is consulted.

- If you want your system to consult NIS if it fails to find a host name in BIND or if the BIND name servers are not responding, create or modify the `hosts` line in the `/etc/nsswitch.conf` file as follows:

  ```
  hosts: dns [NOTFOUND=continue TRYAGAIN=continue] nis files
  ```

  With this configuration, if BIND does not return the requested information, and NIS is not running, the `/etc/hosts` file is consulted.

HP recommends that you maintain at least a minimal `/etc/hosts` file that includes important addresses like gateways, diskless boot servers and root servers, and your host's own IP address. HP also recommends that you include the word `files` in the `hosts` line to help ensure a successful system boot using the `/etc/hosts` file when BIND and NIS are not available.

CAUTION Changing the default configuration can complicate troubleshooting. The default configuration is designed to preserve the authority of the name service you are using. It switches from BIND to NIS only if BIND is not enabled. It switches from NIS to the local `/etc` file only if NIS is not enabled. It is very difficult to diagnose problems when multiple name servers are configured and enabled for use.

For more information on the Name Service Switch, type `man 4 switch` at the HP-UX prompt.

Syntax of the `nsswitch.conf` File

Each line in the `/etc/nsswitch.conf` file has the following syntax:

lookup_type name_service [*status=action status=action* ...] *name_service* ...

If you include any *status=action* pairs after a name service, the square brackets are required.

lookup_type The type of information to be looked up. The supported keywords and the information types they represent are listed in Table 5-1. These keywords are case-sensitive.

name_service One of the following name services to use for the type of information in the *lookup_type* field. These keywords *must* be in lowercase.

- `files` Files in the `/etc` directory on the local host (`/etc/hosts`, `/etc/services`, and so on).
- `nis` Network Information Service (NIS).
- `dns` Domain Name System (DNS), which is implemented by Berkeley Internet Name Domain (BIND) on HP-UX. See the *Installing and Administering Internet Services* manual for more information. The `dns` keyword may be used only on the line beginning with `hosts`.

status One of the following statuses returned by a name service query. These values may be entered in uppercase or lowercase.

- `SUCCESS` The lookup was successful, and the requested information was found.
- `NOTFOUND` The name service returned a response, but the requested data was not in its database.
- `UNAVAIL` The name service is not configured.

`TRYAGAIN` The name service was busy and the request timed out. This value is returned only by DNS.

action The action to take based on the status of the name service query. The following values may be entered in uppercase or lowercase.

`continue` Try the next name service in the list.

`return` End the lookup and return control to the calling process without consulting the next name service in the list.

If a line beginning with one of the l*ookup_type*s does not exist in the `/etc/nsswitch.conf` file, the default Name Service Switch configuration for that type of information is used. If the `/etc/nsswitch.conf` file does not exist, the default configuration is used for every type of information. The default Name Service Switch configuration is described in "Default Configuration" on page 160.

Table 5-1 Types of Lookups Controlled by the Name Service Switch

Keyword	Type of Information Represented by Keyword
`automount`	NFS automounter maps stored in files like `/etc/auto_master` and `/etc/auto_home` or in NIS maps like `auto.master` and `auto.home`.
`hosts`	Host names and IP addresses stored in the `/etc/hosts` file or the NIS `hosts.byaddr` and `hosts.byname` maps.
`netgroup`	NFS netgroups stored in the `/etc/netgroup` file or the NIS `netgroup`, `netgroup.byhost` and `netgroup.byuser` maps.
`networks`	Network names and IP addresses stored in the `/etc/networks` file or the NIS `networks.byaddr` and `networks.byname` maps.
`protocols`	Networking protocol names and numbers stored in the `/etc/protocols` file or the NIS `protocols.byname` and `protocols.bynumber` maps.
`rpc`	RPC program names and numbers stored in the `/etc/rpc` file or the NIS `rpc.byname` and `rpc.bynumber` maps.
`services`	Mapping of networking services to port numbers and protocols, stored in the `/etc/services` file or the NIS `services.byname` and `services.bynp` maps.

Default Configuration

A default `nsswitch.conf` file is supplied in the `/usr/newconfig/etc` directory. It contains the following lines:

```
hosts:     dns  nis  files
protocols: nis  files
services:  nis  files
networks:  nis  files
netgroup:  nis  files
rpc:       nis  files
```

This is the default configuration. In other words, if you copy `/usr/newconfig/etc/nsswitch.conf` to `/etc/nsswitch.conf`, the Name Service Switch behaves the same way it would if no `/etc/nsswitch.conf` file existed.

If your `/etc/nsswitch.conf` file contains a syntactically correct line for a particlar type of information, that line is used instead of the default.

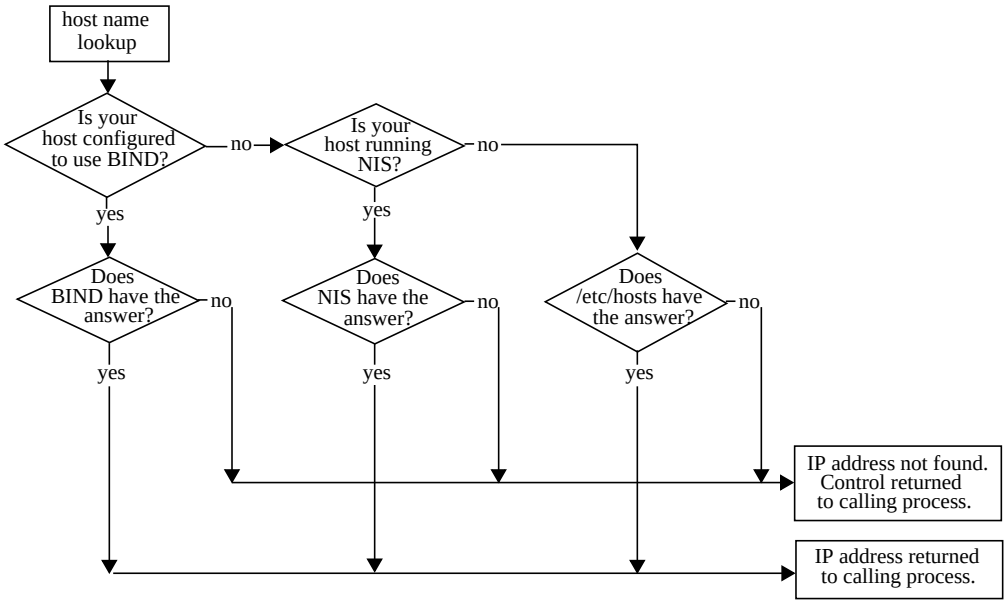
If you specify a name service for a particular type of information, but you do not specify four *status=action* pairs after the name service, the following default *status=action* pairs are used for any statuses you did not specify:

```
SUCCESS=return
NOTFOUND=return
UNAVAIL=continue
TRYAGAIN=return
```

So, for example, in the default configuration for `protocols`, NIS will be consulted first, and if NIS is not configured (the query returns `UNAVAIL`), the local `/etc/protocols` file will be consulted. If the query returns anything other than `UNAVAIL`, the `/etc/protocols` file will not be consulted.

Figure 5-1 illustrates the default behavior of the Name Service Switch for host information lookups.

Figure 5-1 Default Behavior of the Name Service Switch



Troubleshooting the Name Service Switch

This section describes the following methods for troubleshooting your Name Service Switch configuration:

- To Check the Syntax of the hosts Line
- To Check the Current hosts Configuration
- To Trace a Host Name Lookup

To Check the Syntax of the hosts Line

To check the syntax of the `hosts` line in `/etc/nsswitch.conf` file, start `nslookup` with the `swdebug` option, as follows:

```
nslookup -swdebug
```

You will see the output of the parser as it reads the `hosts` line in your `nsswitch.conf` file. If your `hosts` line is syntactically correct, you will see the line `__nsw_getconfig: PARSE SUCCESSFUL`. If your `hosts` line contains a syntax error, you will see the line `__nsw_getconfig: ERR-SYNTAX ERROR`.

The following example checks the syntax of a `hosts` line that is missing a closing square bracket:

```
# cat /etc/nsswitch.conf
hosts: dns [notfound=continue] nis [notfound=continue files

# nslookup -swdebug
__nsw[/etc/nsswitch.conf] 1->hosts: dns [notfound=continue] nis [notf
ound=continue files
__nsw[/etc/nsswitch.conf]LS->L<hosts>L<:>L<dns>L<[>L<notfound>L<=>L<c
ontinue>L<]>L<nis>L<[>L<notfound>L<=>L<continue>L<files>^Missing =^
__nsw.error_recovery: ERR- Error Recovery Completed
__nsw_getconfig: ERR- SYNTAX ERROR
__nsw_getdefault: default hosts lookup policy
Default Name Server:  hpindbu.cup.hp.com
```

The parser indicates the error with carats (^). In this case, the parser reads the word `files` as another status following `notfound=continue`, because it has not encountered a closing square bracket. If the word `files` were a status, it must be followed by an equal sign, and it is not. So the parser displays the message `^Missing =^`.

NOTE The parser checks only the position of the elements with respect to the delimiters `:`, `[`, and `]`. It does not check the spelling of all the elements. For example, if you type `dsn` instead of `dns`, you receive the `PARSE SUCCESSFUL` message. However, when you attempt a host name lookup, `dsn` is not a known name service, so DNS is not queried, and the lookup switches to the next configured source.

To Check the Current `hosts` Configuration

To check the Name Service Switch configuration that your system is currently using for host information, start `nslookup` and issue the `policy` command, as follows:

```
# nslookup
> policy
```

The output for the default configuration is as follows:

```
# Lookups = 3
dns [RRCR]     nis [RRCR]     files [RRRR]
```

The letters in square brackets stand for (R)eturn or (C)ontinue. They represent the values of the four status values, `SUCCESS`, `NOTFOUND`, `UNAVAIL`, and `TRYAGAIN`. In the example, the *status=action* pairs configured for `dns` and `nis` are

- `SUCCESS=return`
- `NOTFOUND=return`
- `UNAVAIL=continue`
- `TRYAGAIN=return`

For the following `hosts` line

```
hosts:  dns [NOTFOUND=continue]  files
```

the `policy` command displays the following:

```
# Lookups = 2
dns [RCCR]     files [RRRR]
```

To stop the `nslookup` program, type `exit`.

To Trace a Host Name Lookup

To trace a host name lookup, start `nslookup`, set the `swtrace` option, and perform a lookup, as follows:

```
# nslookup
> set swtrace
> hostname
```

For the `nsswitch.conf` file containing the `hosts` line

```
hosts: dns [NOTFOUND=continue] nis [NOTFOUND=continue] files
```

the following example tries all three name services before it finds an answer:

```
# nslookup
> set swtrace
> romney
Name Server: hpindbu.cup.hp.com
Address: 15.13.104.13

lookup source is DNS
Name Server: hpindbu.cup.hp.com
Address: 15.13.104.13

*** hpindbu.cup.hp.com can't find romney: Non-existent domain

Switching to next source in the policy
lookup source is NIS
Default NIS Server: hpntc43c
Address: 15.13.119.52
Aliases: hpntc43c.cup.hp.com, hpntc43c-119, 3c-119

*** No address information is available for "romney"

Switching to next source in the policy
lookup source is FILES
Using /etc/hosts on: hpntc2k

Name: romney
Address: 15.13.104.128
```

NOTE If you do not set `swtrace`, `nslookup` displays only the first name service where it looks for a host, even if it finds the host in another name service.

6 Configuring and Using the Remote Execution Facility (REX)

The Remote Execution Facility (REX) allows you to execute commands on a remote host. REX is similar to the `remsh`(1) command, except REX simulates the user's home environment on the remote host and mounts the user's current working directory on the remote host. REX consists of the following:

- The `on` command, which is the user interface to REX and runs on the host where the user is logged in. The host where the `on` command is issued is known as the **REX client**.
- The `rexd` daemon, which runs on the remote host. The host running the **rexd** daemon is known as the **REX server**.

This chapter contains the following sections:

- How REX Works
- Configuring REX

How REX Works

1. A user issues the `on` command, specifying a command to execute and the name of a remote host on which to execute it.

   The user must be logged in as a non-root user (a user with a non-zero user ID) to use the `on` command. Also, an account with the user's local user ID must exist on the remote host.

2. The `on` command passes the user's environment variables to the remote host. If the command is interactive, the `on` command also passes some of the user's `tty` settings to the remote host. Note that the user's environment and `tty` settings on the remote system will not be identical to those on the user's home system.

3. The `rexd` daemon running on the remote host NFS-mounts the user's current working directory on the remote host, if it is not already mounted there.

   By default, `rexd` mounts the user's current working directory under `/var/spool/rexd/rexd`*AXXXX*`/`*current_directory*, where *AXXXX* is a letter followed by a four-digit number, and *current_directory* is the full pathname of the user's current working directory on the local system.

4. The command that the user specified with the `on` command is executed on the remote host (the REX server). If the user did not specify a command to execute, a shell is started on the REX server.

5. After the command has executed on the REX server, `rexd` unmounts the user's current working directory. If the directory is busy, `rexd` will not be able to unmount it.

For more information on REX, type `man 1M rexd` or `man 1 on` at the HP-UX prompt.

REX Example

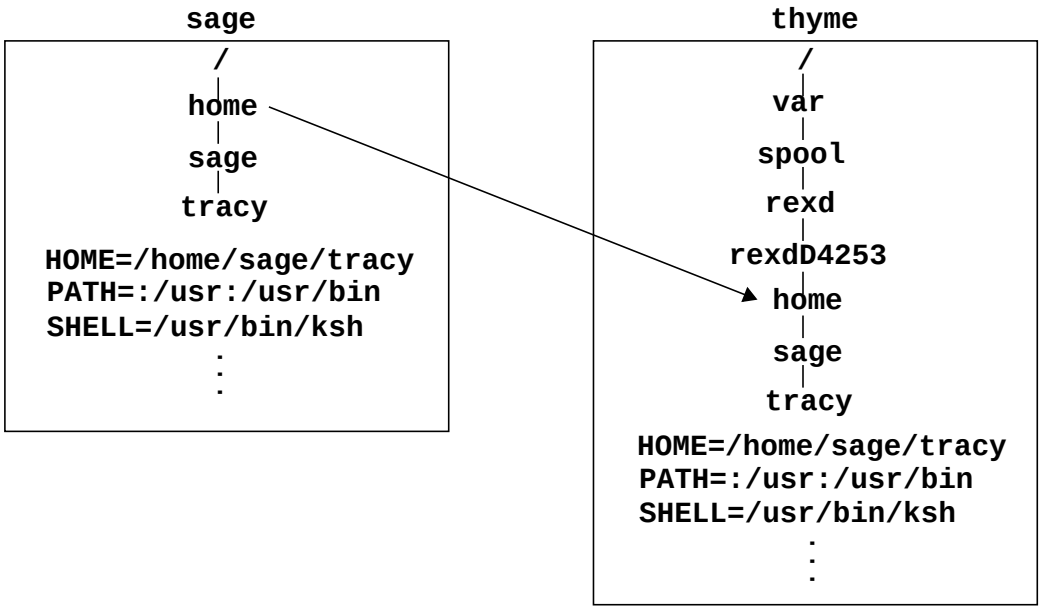
In the following example, user `tracy` is logged into host `sage`. Her current working directory is her home directory, `/home/sage/tracy`. She issues the `on` command to run `more` on host `thyme`:

```
on -i thyme more /etc/exports
```

The `-i` option is required, because `more` is an interactive command.

`tracy`'s home environment on host `sage` is transfered to host `thyme`. `tracy`'s current working directory (her home directory, in this example) is mounted on host `thyme`.

Figure 6-1 REX Example



The `more` command from the `/usr/bin` directory on host `thyme` executes, listing the `/etc/exports` file from host `thyme`. The output of the `more` command is directed to `tracy`'s display on host `sage`.

After `tracy` types `q` to `quit` the `more` program, her current working directory is unmounted from host `thyme`.

Configuring REX

This section tells you how to set up REX clients and REX servers. It also explains how to configure added security for REX servers and how to configure logging for the `rexd` daemon.

To Configure REX

1. Make sure all the hosts to which users need access are listed in your hosts database (BIND, NIS, or `/etc/hosts`).
2. Make sure users have accounts on all the hosts they need to use. Make sure the user ID for each user is the same on all hosts where that user has an account.

   If you are using NIS, and users do not need access to any hosts outside your NIS domain, this step is not necessary. With NIS, user accounts are administered centrally on the NIS master server, and all hosts have access to the same user information. See "Configuring and Administering NIS" on page 101 for instructions on setting up NIS.
3. Make sure all REX clients (hosts from which users will issue the `on` command) are configured as NFS servers. See "Configuring and Administering an NFS Server" on page 22.
4. Make sure users' home directories on all REX clients are exported to REX servers (available to be mounted with NFS). See "To Make Directories Available to NFS Clients (Export Directories)" on page 23.
5. Make sure all REX servers (hosts where the `rexd` daemon will run) are configured as NFS clients. See "Configuring and Administering an NFS Client" on page 33.
6. Use a text editor to uncomment the following line in the `/etc/inetd.conf` file, which starts `rexd`:

   ```
   rpc stream tcp nowait root /usr/sbin/rpc.rexd 100017 1 rpc.rexd
   ```
7. Issue the following command to force `inetd` to reread its configuration file:

   ```
   /usr/sbin/inetd -c
   ```

To Configure REX Security

1. On each REX server, add the `-r` option to the line in `/etc/inetd.conf` that starts the `rexd` daemon, as follows:

   ```
   rpc stream tcp nowait root /usr/sbin/rpc.rexd 100017 1 \
   rpc.rexd -r
   ```

2. Issue the following command to force `inetd` to reread `/etc/inetd.conf`:

   ```
   /usr/sbin/inetd -c
   ```

3. Add lines to the `/etc/hosts.equiv` file on the REX server to allow REX clients to use the server,

   or

   have each REX user add lines to a `.rhosts` file in the user's home directory on the REX server to allow access from REX clients.

The `-r` option causes `rexd` to deny requests from a user on a REX client unless the client is listed in `/etc/hosts.equiv` or the user's `$HOME/.rhosts` file on the REX server.

A line in the `/etc/hosts.equiv` or `$HOME/.rhosts` file has the following syntax:

`hostname [`*`username`*`]`

For example, if user `paula` has accounts on REX clients `broccoli` and `cabbage` and on REX server `cauliflower`, she would create a `.rhosts` file in her home directory on `cauliflower` with the following lines:

```
broccoli  paula
cabbage   paula
```

CAUTION

The `/etc/hosts.equiv` and `$HOME/.rhosts` files create a significant security risk. Make sure these files and users' home directories are writable only by the owner.

For more information, see the man pages for `rexd`(1M) and `hosts.equiv`(4).

To Configure Logging for the rexd Daemon

1. Use a text editor to add the `-l` *log_file* option to the line in `/etc/inetd.conf` that starts `rexd`, as in the following example:

   ```
   rpc stream tcp nowait root /usr/sbin/rpc.rexd 100017 1  \
   rpc.rexd -l /var/adm/rexd.log
   ```

2. Issue the following command to force `inetd` to reread its configuration file:

   ```
   /usr/sbin/inetd -c
   ```

When logging is turned on, `rexd` logs any diagnostic, warning, and error messages to *log_file*. If *log_file* exists, `rexd` appends messages to the file. If *log_file* does not exist, `rexd` creates it. Messages are not logged if the `-l` option is not specified.

Information logged to the file includes date and time of the error, host name, process ID and name of the function generating the error, and the error message.

Different RPC services can share a single log file, because enough information is included to uniquely identify each error.

Type `man 1M rexd` for explanations of the messages logged by the `rexd` daemon.

Many of the errors logged by `rexd` are also returned to the user who issued the `on` command. Type `man 1 on` for explanations of the messages returned by the `on` command.

7 Troubleshooting NFS Services

This chapter describes tools and procedures for troubleshooting the NFS Services. It contains the following sections:

- Common Problems with NFS
- Common Problems with NIS
- Performance Tuning
- Logging and Tracing of NFS Services
- Normal System Startup

Common Problems with NFS

This section lists the folllowing common problems encountered with NFS and suggests ways to correct them.

If You Receive an NFS "Server Not Responding" Message

- Issue the `/usr/sbin/ping`(1M) command on the NFS client to make sure the NFS server is up and is reachable on the network. If the `ping` command fails, either the server is down, or the network has a problem. If the server is down, reboot it, or wait for it to come back up. For information on troubleshooting network problems, see *Installing and Administering LAN/9000 Software.*
- Issue the following command on the NFS client to make sure the server is running all the NFS server processes:

  `/usr/bin/rpcinfo -p` *servername*

  The `rpcinfo` command should display the following processes:

  - `portmap`
  - `nfs`
  - `mountd`
  - `status`
  - `nlockmgr`
  - `llockmgr`

  If any of these processes is not running, follow these steps:

  1. Make sure the `/etc/rc.config.d/nfsconf` file on the NFS server contains the following lines:

     ```
     NFS_SERVER=1
     START_MOUNTD=1
     ```

  2. Make sure that the `/etc/inetd.conf` file on the NFS server does *not* contain a line to start `rpc.mountd`. If it does, make sure the `START_MOUNTD` variable in `/etc/rc.config.d/nfsconf` is set to 0.
  3. Issue the following command on the NFS server to start all the necessary NFS processes:

     ```
     /sbin/init.d/nfs.server start
     ```

- Issue the following command on the NFS client to make sure the `rpc.mountd` process on the NFS server is available and responding to RPC requests:

```
/usr/bin/rpcinfo -u servername mountd
```

If the `rpcinfo` command returns `RPC_TIMED_OUT`, the `rpc.mountd` process may be hung. Issue the following commands on the NFS server to restart `rpc.mountd` (*PID* is the process ID returned by the `ps` command):

```
/usr/bin/ps -ef | /usr/bin/grep mountd
/usr/bin/kill PID
/usr/sbin/rpc.mountd
```

- You can receive "server not responding" messages when the server or network is heavily loaded and the RPC requests are timing out. Try doubling the `timeo` mount option for the directory, as in the following example from the `/etc/fstab` file, which changes the `timeo` value from 7 (the default) to 14. (The `timeo` option is in tenths of a second.)

  ```
  cabbage:/usr /usr nfs nosuid,timeo=14 0 0
  ```

- Issue the following command on the NFS client to check that your `hosts` database returns the correct address for the NFS server:

  ```
  /usr/bin/nslookup server_name
  ```

  If your client cannot resolve the server's hostname, see "If You Receive an "Unknown Host" or "Not In Hosts Database" Message" on page 182.

  Issue the same `nslookup` command on the NFS server, and compare the address with the one returned by the `nslookup` command on the NFS client. If they are different, correct your NIS, BIND, or `/etc/hosts` configuration. For information on NIS troubleshooting, see "Common Problems with NIS" on page 192. For information on BIND or `/etc/hosts`, see *Installing and Administering Internet Services*.

- If you are using AutoFS, issue the `ps -ef` command to make sure the `automountd` process is running on your NFS client. If it is not, follow these steps:

  1. Make sure the `AUTOMOUNT` variable is set to 1 in the `/etc/rc.config.d/nfsconf` file on the NFS client.

     ```
     AUTOMOUNT=1
     ```

  2. Issue the following command on the NFS client to start the automounter:

     ```
     /sbin/init.d/nfs.client start
     ```

- If the “server not responding” message was followed by `RPC_AUTH_ERROR; why=AUTH_BOGUS_CREDENTIAL`, this could mean that you (or the user who received the message) are a member of too many groups. On HP-UX release 9.0 or later, you can be a member of up to 16 groups. On HP-UX releases prior to 9.0, you can be a member of up to 8 groups.

If You Receive an "Access Denied" Message

- Issue the following command on the NFS client to check that the NFS server is exporting the directory you want to mount:

  `/usr/sbin/showmount -e` *server_name*

  If the server is not exporting the directory, edit the `/etc/exports` file on the server so that it allows your NFS client access to the directory. Then, issue the following command to force the server to read its `/etc/exports` file.

  `/usr/sbin/exportfs -a`

  If the directory is exported with the `access` option, make sure your NFS client is included in the `access` list, either individually or as a member of a netgroup.

- If your NFS client is included in the `access` list as a member of a netgroup, make sure it is a member of the netgroup in the server's `/etc/netgroup` file.

  If you are using NIS to manage your netgroups, issue the following command to determine whether your NIS server has up-to-date information about the netgroup that includes your client:

  `/usr/bin/ypmatch` *netgroup_name* `netgroup`

  If your NIS server does not return the correct information, see "Common Problems with NIS" on page 192.

- Issue the following commands on the NFS server to make sure your NFS client is listed in its `hosts` database:

  `nslookup` *client_name*
  `nslookup` *client_IP_address*

  If the server cannot resolve your client's hostname, see "If You Receive an "Unknown Host" or "Not In Hosts Database" Message" on page 182.

- If `rpc.mountd` is configured in `/etc/inetd.conf` on the NFS server, check the server's `/var/adm/inetd.sec` file to make sure your NFS client is allowed access to `rpc.mountd`.

If You Receive a "Permission Denied" Message

- Check the mount options in the `/etc/fstab` file on the NFS client. A directory you are attempting to write to may have been mounted read-only.
- Issue the `ls -l` command to check the HP-UX permissions on the server directory and on the client directory that is the mount point. You may not be allowed access to the directory.
- Issue the following command on the NFS server:

  `/usr/sbin/exportfs`

  Or, issue the following command on the NFS client:

  `/usr/sbin/showmount -e` *server_name*

  Check the export permissions on the exported directory. The directory may have been exported read-only to your client. The system administrator of the NFS server can use the `remount` mount option to mount the directory read/write without unmounting it. See "To Change the Default Mount Options" on page 40.

  If you are logged in as root to the NFS client, check the export permissions to determine whether root access to the directory is granted to your NFS client.
- If you are logged in as root to the NFS client, and your client is not allowed root access to the exported directory, check the `passwd` database on the NFS server to determine whether it contains an entry for user `nobody`. Without root access, the root user on an NFS client is given the access permissions of user `nobody`. Also, check whether anonymous users are denied access to the directory (with the `anon=65535` export option).

  If your client is not allowed root access or anonymous user ID access to the exported directory, log in as a non-root user to get access to the directory.
- If you are not running NIS, or if the server is in a different domain from the client, check the `passwd` databases on the server and the client to make sure you have a valid login on both machines and that your user ID is the same on both machines. If your user ID is unrecognized on the NFS server, you will be granted the permissions of user `nobody`.

- If you were attempting to run a program when you received the "permission denied" message, issue the `ls -l` command on the NFS server to check whether the program you tried to run has the setuid bit set. If it does, check `/etc/fstab` to determine whether the directory was mounted with the `nosuid` mount option. If necessary, remove the `nosuid` option from the `/etc/fstab` file, then unmount and remount the directory.

If You Receive an "Unknown Host" or "Not In Hosts Database" Message

❑ Issue the following commands to trace a lookup of the unknown host:

```
nslookup
> set swtrace
> hostname
```

The trace will indicate which name services (BIND, NIS, or `/etc/hosts`) were queried and in what order. If your host is not performing lookups the way you want, see "Configuring the Name Service Switch" on page 153 for instructions on configuring the Name Service Switch.

Type `exit` to exit from `nslookup`.

❑ If your host is using the `/etc/hosts` file to resolve hostnames, edit the file to add or correct the entry for the unknown host. Type `man 4 hosts` for the correct syntax.

❑ If your host is using NIS to resolve hostnames, see "Common Problems with NIS" on page 192.

❑ If your host is using BIND (DNS) to resolve hostnames, see *Installing and Administering Internet Services* for instructions on troubleshooting BIND.

If You Receive a "Device Busy" Message

- If you received the "device busy" message while attempting to mount a directory, try to access the mounted directory. If you can access it, then it is already mounted.
- If you received the "device busy" message while attempting to unmount a directory, a user or process is currently using the directory. Wait until the process completes, or follow these steps:

  1. Issue the following command to determine who is using the mounted directory:

     `/usr/sbin/fuser -cu` *local_mount_point*

     The `fuser`(1M) command will return a list of process IDs and user names that are currently using the directory mounted under *local_mount_point*. This will help you decide whether to kill the processes or wait for them to complete.

  2. To kill all processes using the mounted directory, issue the following command:

     `/usr/sbin/fuser -ck` *local_mount_point*

  3. Try again to unmount the directory.

If You Receive a "Stale File Handle" Message

A "stale file handle" occurs when one client removes an NFS-mounted file or directory that another client has open, as in the following sequence of events:

	NFS client 1	NFS client 2
1	`% cd /proj1/source`	
2		`% cd /proj1`
3		`% rm -Rf source`
4	`% ls` `.:Stale File Handle`	

If a server stops exporting a directory that a client has mounted, the client will receive a stale file handle error. Stale file handles also occur if you restore the NFS server's file systems from a backup or randomize the inode numbers with `fsirand`(1M).

- If the stale file handle occurred because someone removed a file or directory that was in use, or because a server stopped exporting a directory that was in use, follow these steps:

  1. Issue the `/usr/bin/cd` command to move out of the NFS-mounted directory that is causing the problem, then try unmounting the directory:

     ```
     /usr/bin/cd ..
     /usr/sbin/umount directory
     ```

  2. If the directory cannot be unmounted because it is busy (in use), issue the following commands to kill the processes using the directory and to try again to unmount it:

     ```
     /usr/sbin/fuser -ck local_mount_point
     /usr/sbin/umount local_mount_point
     ```

  3. If the directory still cannot be unmounted, reboot the client.

  4. To avoid stale file handles caused by users deleting NFS-mounted files, try using a source code control system, like Revision Control System (RCS). A source code control system allows only one user

at a time to modify a file or directory, so one user cannot remove files another user is accessing. Type `man 5 rcsintro` for more information.

- If someone has restored the server's file systems from backup or issued the `fsirand` command on the server, follow these steps on each of the NFS clients to prevent stale file handles by restarting NFS:

  1. Issue the `mount`(1M) command with no options, to get a list of all the mounted file systems on the client:

     ```
     /usr/sbin/mount
     ```

  2. For every NFS-mounted directory listed by the `mount` command, issue the following command to determine whether the directory is currently in use:

     `/usr/sbin/fuser -cu` *local_mount_point*

     This command lists the process IDs and user names of everyone using the mounted directory.

  3. Warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can use the following command to kill all processes using the directory:

     `/usr/sbin/fuser -ck` *local_mount_point*

  4. Issue the following command on the client to unmount all NFS-mounted directories:

     ```
     /usr/sbin/umount -at nfs
     ```

  5. Issue the following commands to restart the NFS client:

     ```
     /sbin/init.d/nfs.client stop
     /sbin/init.d/nfs.client start
     ```

If a Program Hangs

- ❏ Check whether the NFS server is up and operating correctly. See "If You Receive an NFS "Server Not Responding" Message" on page 176.

  If the server is down, wait until it comes back up, or, if the directory was mounted with the `intr` mount option (the default), you can interrupt the NFS mount, usually with `CTRL-C`.

- ❏ If the program uses file locking, issue the following commands (on either the client or the server) to make sure `rpc.statd` and `rpc.lockd` are available and responding to RPC requests:

  ```
  /usr/bin/rpcinfo -u servername status
  /usr/bin/rpcinfo -u servername llockmgr
  /usr/bin/rpcinfo -u servername nlockmgr
  /usr/bin/rpcinfo -u clientname status
  /usr/bin/rpcinfo -u clientname llockmgr
  /usr/bin/rpcinfo -u clientname nlockmgr
  ```

  If any of these commands returns `RPC_TIMED_OUT`, the `rpc.statd` or `rpc.lockd` process may be hung. Follow these steps to restart `rpc.statd` and `rpc.lockd`:

  1. Issue the following commands, on both the NFS client and the NFS server, to kill `rpc.statd` and `rpc.lockd` (*PID* is a process ID returned by the `ps` command):

     ```
     /usr/bin/ps -ef | /usr/bin/grep rpc.statd
     /usr/bin/kill PID
     /usr/bin/ps -ef | /usr/bin/grep rpc.lockd
     /usr/bin/kill PID
     ```

  2. Issue the following commands, on both the client and the server, to remove the contents of the `sm` and `sm.bak` directories:

     ```
     /usr/bin/rm -r /etc/sm
     /usr/bin/rm -r /etc/sm.bak
     ```

  3. Issue the following commands to restart `rpc.statd` and `rpc.lockd` on both the client and the server:

     ```
     /usr/sbin/rpc.statd
     /usr/sbin/rpc.lockd
     ```

NOTE Always start `rpc.statd` before starting `rpc.lockd`.

  4. Issue the following commands to verify that `rpc.statd`, `rpc.lockd`, and `nfsd` are all running and responding to RPC requests:

```
/usr/bin/rpcinfo -u servername status
/usr/bin/rpcinfo -u servername llockmgr
/usr/bin/rpcinfo -u servername nlockmgr
/usr/bin/rpcinfo -u servername nfs
/usr/bin/rpcinfo -u clientname status
/usr/bin/rpcinfo -u clientname llockmgr
/usr/bin/rpcinfo -u clientname nlockmgr
/usr/bin/rpcinfo -u clientname nfs
```

5. Wait two minutes before retrying the mount that caused the program to hang.
6. If the problem persists, restart `rpc.statd` and `rpc.lockd`, and turn on tracing. See “To Start and Stop Detailed Logging of rpc.statd and rpc.lockd” on page 210 and “To Start and Stop Basic Logging of rpc.statd and rpc.lockd” on page 211.

If Data is Lost Between the Client and the Server

- Make sure the directory is exported from the server with the `noasync` option (the default). If the directory is exported with the `async` option, the NFS server will acknowledge NFS writes before actually writing data to disk. Changing an exported directory from `async` to `noasync` degrades write performance for that directory.
- If users or applications will be writing to the NFS-mounted directory, make sure it is mounted with the `hard` option (the default), rather than the `soft` option.
- If you have a small number of NFS applications that require absolute data integrity, add the `O_SYNC` flag to the `open()` calls in your applications. When you open a file with the `O_SYNC` flag, a `write()` call will not return until the write request has been sent to the NFS server and acknowledged. The `O_SYNC` flag degrades write performance for applications that use it.
- If you have a large number of NFS applications requiring absolute data integrity, or if your entire installation needs a high degree of data integrity, set the `NUM_NFSIOD` variable to 0 in the `/etc/rc.config.d/nfsconf` file on each client, as follows,

  ```
  NUM_NFSIOD=0
  ```

  and issue the following commands to kill all the `biod` processes (*PID* is a process ID returned by the `ps` command):

  ```
  /usr/bin/ps -ef | /usr/bin/grep biod
  /usr/bin/kill PID PID ...
  ```

  The `biod` daemons improve performance by handling NFS read and write requests from users and applications. After a write request is passed to a `biod` daemon, control is returned to the user or application. Running a client without `biod` daemons degrades NFS performance for all users and applications on that client.
- If multiple NFS users will be writing to the same file, add the `lockf()` call to your applications to lock the file so that only one user may modify it at a time.

  If multiple users on different NFS clients will be writing to the file, you must also turn off attribute caching on those clients by mounting the file with the `noac` mount option. Turning off attribute caching degrades NFS performance.

For more information, see the following man pages: `mount`(1M), `open`(2), `write`(2), `lockf`(2), and `biod`(1M).

If You Cannot Start New Processes

- Issue the following command to check your server's memory utilization:

  ```
  netstat -m
  ```

  If the number of `requests for memory denied` is high, your server does not have enough memory. Consider adding more memory or using a different host as the NFS server.

- Issue the `ps -ef` command on the NFS server, and check for many instances of the same application. Sometimes an application clones itself indefinitely until it uses up all the available inodes on a system.

- The default maximum number of inodes shipped with HP-UX tends to be too small for sites that make extensive use of NFS. Follow this procedure to increase the maximum number of inodes on your NFS server:

  1. Log in as root to the NFS server.
  2. Type `/usr/sbin/sam` to start SAM (System Administration Manager).
  3. Open `Kernel Configuration`.
  4. Open `Configurable Parameters`.
  5. Highlight the line that begins with `ninode`, and choose `Modify Configurable Parameter` from the `Actions` menu.
  6. Increase the value in the `Formula/Value` field, either by changing the constant multiplier in the formula or replacing the formula with a value. If your `ninode` value is currently set to the default (606), try changing it to 2048.
  7. Use SAM to regenerate the kernel and reboot the system.

For more information on using SAM, choose SAM's `Help` button, or press the `F1` key for context-sensitive help.

If You Receive a "Too Many Levels of Remote in Path" Message

This message indicates that you are attempting to mount a directory from a server that has NFS-mounted the directory from another server. You cannot "chain" your NFS mounts this way. You must mount the directory from the server that has it mounted on a local disk.

Common Problems with NIS

This section lists the following common problems encountered with NIS and suggests ways to correct them.

If You Receive an NIS "Server Not Responding" Message

- Issue the `/usr/sbin/ping`(1M) command on the NIS client to make sure the NIS server is up and is reachable on the network. If the `ping` command fails, either the server is down, or the network has a problem. If the server is down, reboot it, or wait for it to come back up. For information on troubleshooting network problems, see *Installing and Administering LAN/9000 Software*.

  To boot your NIS client without waiting for the server to come up, boot the client in single user mode, set `NIS_CLIENT=0` in the `/etc/rc.config.d/namsvrs` file, then boot your client the rest of the way up.

- Issue the `domainname` command (with no arguments) on both the NIS server and the NIS client to check whether their domain names are the same. If they are different, log in as root to the NIS client and issue the following command to change its domain name:

  `domainname` *domainname*

- Issue the `ps -ef` command on the NIS server to check whether `ypserv` is running. If it is not, follow these steps:

  1. In the `/etc/rc.config.d/namesvrs` file on the NIS server, make sure the following variables are set:

     ```
     NIS_MASTER_SERVER=1
     ```

  2. Issue the following command to start up the NIS server:

     ```
     /sbin/init.d/nis.server start
     ```

- Make sure an NIS server exists on the same subnet as the NIS client. The client broadcasts its bind request, and it binds to the first server that responds to the request. Broadcasts do not cross gateways or routers, so the server must be on the same subnet as the client in order to receive the bind request. If you cannot configure an NIS server on the same subnet as your NIS clients, see "To Bind an NIS Client to a Server on a Different Subnet" on page 144.

If a User Cannot Log In

- If the user has recently changed passwords, ask the user to try logging in with the old password. If the user can log in using the old password, follow these steps:

  1. Issue the `ps -ef` command on the NIS master server to make sure the `yppasswdd` daemon is running. If it is not, issue the following command to start all the NIS server processes:

     ```
     /sbin/init.d/nis.server start
     ```

  2. Check the `cron` scripts on the slave servers to make sure transfers of the `passwd` map from the master server are frequent enough. Once per hour is usually frequent enough, but frequent map transfers may cause too much network traffic. You might want to schedule map transfers for late at night, and advise users to make their password changes just before they go home.

- Issue the following command on the NIS client to determine which master server supplies the `passwd` map to the client:

  ```
  /usr/bin/ypwhich -m passwd
  ```

  If the server does not respond, see "If You Receive an NIS "Server Not Responding" Message" on page 193.

  If the `ypwhich` command returns the name of the NIS master server, log in as root to the master server and make sure the user has an entry in its `/etc/passwd` file. Then, issue the following commands on the master server to generate the NIS `passwd` database from the `/etc/passwd` file and push it to the NIS slave servers:

  ```
  cd /var/yp
  /usr/ccs/bin/make passwd
  ```

- Issue the `domainname` command (with no arguments) to make sure the client's default domain is the domain served by the NIS master server. If it is not, log in as root to the NIS client, and issue the following command to change its domain name:

  `domainname` *`domainname`*

- Issue the following command to check whether the NIS client has an entry in the `passwd` database on the NIS server to which it is bound:

  `/usr/bin/ypmatch` *`username`* `passwd`

  If the client has no entry in the `passwd` database, issue the following command on the NIS server to which the client is bound:

```
/usr/sbin/ypxfr passwd
```

This command transfers the `passwd` database from the NIS master server to the server where you issue the command.

- If the user's NIS client is bound to a slave server, make sure the slave server is listed in the NIS master server's `ypservers` database. Follow these steps:

  1. Issue the following command on the NIS client to determine which server the client is bound to:

     ```
     /usr/bin/ypwhich
     ```

  2. Log into the NIS master server, and issue the following command:

     ```
     cd /var/yp/domainname
     ```

  3. Issue the following command on the NIS master server to write the contents of the `ypservers` database to a temporary file:

     ```
     /usr/sbin/makedbm -u ypservers > tempfile
     ```

  4. If the NIS slave server is not listed in `tempfile`, use a text editor to add it, and then issue the following command to rebuild the `ypservers` database:

     ```
     /usr/sbin/makedbm tempfile ypservers
     ```

- Make sure the NIS escape entry in the `/etc/passwd` file on the client does *not* have an asterisk in the password field. On HP systems, the NIS escape entry in the `/etc/passwd` file should be

  ```
  +::-2:60001:::
  ```

If You Receive an "Unknown Host" Message

❑ Issue the following commands to trace a lookup of the unknown host:

```
nslookup
> set swtrace
> hostname
```

The trace will indicate which name services (BIND, NIS, or `/etc/hosts`) were queried and in what order. If your host is not performing lookups the way you want, see "Configuring the Name Service Switch" on page 153 for instructions on configuring the Name Service Switch.

Type `exit` to exit from `nslookup`.

❑ Issue the following command on the NIS client to determine which master server supplies the `hosts` map:

```
/usr/bin/ypwhich -m hosts
```

If the server does not respond, see "If You Receive an NIS "Server Not Responding" Message" on page 193.

If the `ypwhich` command returns the name of the NIS master server, log in as root to the master server and make sure the unknown host is listed in its `/etc/hosts` file. Then, issue the following commands on the master server to generate the NIS `hosts` database from the `/etc/hosts` file and push it to the NIS slave servers:

```
cd /var/yp
/usr/ccs/bin/make hosts
```

❑ Issue the `domainname` command (with no arguments) to make sure the client's default domain is the domain served by the NIS master server. If it is not, log in as root to the NIS client and issue the following command to change its domain name:

```
domainname domainname
```

❑ Issue the following command to check whether the unknown host is listed in the `hosts` database on the NIS server to which the client is bound:

```
/usr/bin/ypmatch hostname hosts
```

If the host is not listed in the `hosts` database, issue the following command on the NIS server to which the client is bound:

```
/usr/sbin/ypxfr hosts
```

This command transfers the `hosts` database from the NIS master server to the server where you issue the command.

- If the NIS client is bound to a slave server, make sure the slave server is listed in the NIS master server's `ypservers` database. Follow these steps:

  1. Issue the following command on the NIS client to determine which server the client is bound to:

     ```
     /usr/bin/ypwhich
     ```

  2. Log in as root to the NIS master server and issue the following command to change to the directory where the domain databases reside:

     ```
     cd /var/yp/domainname
     ```

  3. On the NIS master server, issue the following command to write the contents of the `ypservers` database to a temporary file:

     ```
     /usr/sbin/makedbm -u ypservers > tempfile
     ```

  4. If the NIS slave server is not listed in `tempfile`, use a text editor to add it, and then issue the following command to rebuild the `ypservers` database:

     ```
     /usr/sbin/makedbm tempfile ypservers
     ```

If an NIS Client Cannot Bind to a Server

If NIS commands return any of the following messages,

```
ypcat: can't bind to an NIS server for domain domainname
```

```
ypmatch: can't match key.
         reason: can't communicate with ypbind
```

```
ypwhich: clntudp_create error RPC_PROG_NOT_REGISTERED
```

then `ypbind` is not running on the client. Issue the following command to start all the NIS client processes:

```
/sbin/init.d/nis.client start
```

If NIS Returns Incorrect Information

- Issue the following command on the NIS client to determine which master server supplies the appropriate NIS map:

  `/usr/bin/ypwhich -m` *mapname*

  If the server does not respond, see "If You Receive an NIS "Server Not Responding" Message" on page 193.

- Log in as root to the NIS master server, and issue the following command to check the contents of the appropriate NIS map:

  `/usr/bin/ypcat -k` *mapname*

  If the map contents are not correct, edit the ASCII file from which the map is generated. Then issue the following commands to regenerate the map and push it to the slave servers:

  ```
  cd /var/yp
  /usr/ccs/bin/make mapname
  ```

- Issue the `domainname` command (with no arguments) to make sure the client's default domain is the domain served by the NIS master server. If it is not, log in as root to the NIS client, and issue the following command to change its domain name:

  `domainname` *domainname*

- Issue the following command on the NIS client to check the contents of the map on the NIS server to which the client is bound:

  `/usr/bin/ypcat -k` *mapname*

  If the contents are not correct, log in as root to the server that serves the NIS client, and issue the following command:

  `/usr/sbin/ypxfr` *mapname*

  This command transfers the map from the NIS master server to the server where you issue the command.

- If the NIS client is bound to a slave server, make sure the slave server is listed in the NIS master server's `ypservers` database. Follow these steps:

  1. Issue the following command on the NIS client to determine which server the client is bound to:

     `/usr/bin/ypwhich`

2. Log in as root to the NIS master server and issue the following command to change to the directory where the domain databases reside:

   ```
   cd /var/yp/domainname
   ```

3. On the NIS master server, issue the following command to write the contents of the `ypservers` database to a temporary file:

   ```
   /usr/sbin/makedbm -u ypservers > tempfile
   ```

4. If the NIS slave server is not listed in `tempfile`, use a text editor to add it, and then issue the following command to rebuild the `ypservers` database:

   ```
   /usr/sbin/makedbm tempfile ypservers
   ```

- ❑ Make sure the slave servers have `cron` scripts that schedule regular updates of the map.

Performance Tuning

This section gives suggestions for identifying performance problems in your network and improving NFS performance on your servers and clients. It contains the following sections:

To Diagnose NFS Performance Problems

1. Issue the following command on several of your NFS clients:

   ```
   nfsstat -rc
   ```

2. If the `timeout` and `retrans` values displayed by `nfsstat -rc` are high, but the `badxid` value is close to zero, packets are being dropped before they get to the NFS server.

   Try decreasing the values of the `wsize` and `rsize` mount options to 4096 or 2048 on the NFS clients. See "To Change the Default Mount Options" on page 40.

   See *Installing and Administering LAN/9000 Software* for information on troubleshooting LAN problems.

3. If the `timeout` and `badxid` values displayed by `nfsstat -rc` are of the same magnitude, your server is probably slow. Client RPC requests are timing out and being retransmitted before the NFS server has a chance to respond to them.

   See "To Improve NFS Server Performance" on page 203.

   Try doubling the value of the `timeo` mount option on the NFS clients. See "To Change the Default Mount Options" on page 40.

4. Issue the following command on any machine on the network:

   ```
   netstat -i
   ```

   The number of collisions (`Coll`) divided by the number of output packets (`Opkts`) is the collision rate. If your collision rate is greater than 10%, consider dividing your network into smaller segments and putting an NFS server on each segment. See *Installing and Administering LAN/9000 Software* for information on dividing your network.

To Improve NFS Server Performance

- ❑ Issue the following command to check your server's memory utilization:

  ```
  netstat -m
  ```

  If the number of `requests for memory denied` is high, your server does not have enough memory, and NFS clients will experience poor performance. Consider adding more memory or using a different host as the NFS server.

- ❑ Put heavily used directories on different disks on your NFS servers so they can be accessed in parallel.

- ❑ Make sure your server is running the correct number of `nfsd` processes. See "To Adjust the Number of nfsd Processes" on page 205.

- ❑ Issue the following command on the NFS server:

  ```
  vmstat -n
  ```

  If the `us` and `sy` values under `cpu` are high, and the `id` (idle time) value under `cpu` is close to zero, your server's CPU is heavily loaded. Try using a faster machine as your NFS server. Do not use a gateway or a terminal server as an NFS or NIS server.

- ❑ Issue the following command to determine which processes are using the most CPU:

  ```
  /usr/bin/top
  ```

  The `top` program sorts the processes running on your system, with the most CPU-intensive process at the top of the display. It refreshes the display every five seconds. Try taking some CPU-intensive processes off the server.

  Type `q` to exit the `top` program.

- ❑ Log into the NFS server and issue the following command:

  ```
  nfsstat -s
  ```

  If the number of `readlink` calls is of the same magnitude as the number of `lookup` calls, you have a symbolic link in a file system that is frequently traversed by NFS clients.

  On the NFS clients that require access to the linked directory, mount the target of the link. Then, remove the link from the exported directory on the server.

When a client requests access to a linked file or directory, two requests are sent to the server: one to look up the path to the link, and another to look up the target of the link. You can improve NFS performance by removing symbolic links from exported directories.

CAUTION

Do not remove symbolic links in an NFS diskless environment. File sharing in NFS diskless is done by means of symbolic links.

- If the value of `getattr` displayed by `nfsstat -s` is greater than 60%, one or more clients have either turned off attribute caching (with the `noac` mount option) or set the caching timeout values too low.

  Increase the attribute caching timeouts on the clients that have them set below the default values. See "To Change the Default Mount Options" on page 40.

- Export directories with the `async` option. When `async` is specified, the server acknowledges write requests from clients before writing data to disk. Clients do not have to wait for a write request to complete before issuing another request.

To Adjust the Number of nfsd Processes

1. Issue the following command on the NFS server:

   ```
   netstat -s
   ```

   If the UDP statistics displayed by the `netstat` command indicate a large number of socket overflows, as in the following example, then your server is not running enough `nfsd` daemons.

   ```
   udp:
      0 incomplete headers
      0 bad data length fields
      0 bad checksums
      1375 socket overflows
   ```

2. To increase the number of `nfsd` daemons running, change the value of the `NUM_NFSD` variable in the `/etc/rc.config.d/nfsconf` file, as in the following example:

   ```
   NUM_NFSD=8
   ```

3. Issue the following command to start more `nfsd` processes:

   `/usr/sbin/nfsd` *number*

4. Issue the `netstat -s` command again to check the number of socket overflows. Continue to adjust the `NUM_NFSD` value and start `nfsd` processes until the number of *new* socket overflows is close to zero. (The output of `nfsstat` is cumulative, so when there are no new socket overflows, the number will stay the same.)

As a general rule, an NFS server should run approximately two `nfsd` daemons for each entry in the `/etc/exports` file.

For more information, type `man 1M nfsd` at the HP-UX prompt.

To Improve NFS Client Performance

- Issue the `ps -ef` command to make sure four `biod` processes are running on each client. To start four `biod` processes, set the `NUM_NFSIOD` variable to 4 in the `/etc/rc.config.d/nfsconf` file, and issue the following command:

  ```
  /usr/sbin/biod 4
  ```

NOTE If your performance bottleneck is a slow server, increasing the number of `biod` processes beyond four will not improve NFS performance, and it might make it worse.

- For files and directories that are mounted read-only and never change, set the `actimeo` mount option to 120 or greater in the `/etc/fstab` file on your NFS clients. See "To Change the Default Mount Options" on page 40.
- If you see several "server not responding" messages within a few minutes, try doubling the value of the `timeo` mount option in the `/etc/fstab` file on your NFS clients. See "To Change the Default Mount Options" on page 40.
- If you frequently see the following message when attempting access to a soft-mounted directory,

  `NFS` *operation* `failed for server` *servername*`: Timed out`

  try increasing the value of the `retrans` mount option in the `/etc/fstab` file on the NFS clients. Or, change the soft mount to an interruptible hard mount, by specifying the `hard` and `intr` options (the defaults). See "To Change the Default Mount Options" on page 40.
- Type the following command on the NFS server, to find out the block size of the server's file system:

  `/usr/sbin/tunefs -v` *devicefilename*

  On the NFS clients, set the `wsize` and `rsize` mount options to the `bsize` value displayed by `tunefs`. See "To Change the Default Mount Options" on page 40.
- On the NFS clients, look in the `/etc/fstab` file for "stepping-stone" mounts (hierarchical mounts), as in the following example:

  ```
  thyme:/usr /usr nfs defaults 0 0
  basil:/usr/share /usr/share nfs defaults 0 0
  sage:/usr/share/lib /usr/share/lib nfs defaults 0 0
  ```

Wherever possible, change these “stepping-stone” mounts so that whole directories are mounted from a single NFS server.

Stepping-stone (hierarchical) mounts, like the one in the example above, cause more NFS requests than mounts from a single server. In the example, if a client wants access to something in `/usr/share/lib`, a request must be sent to server `thyme`, then to server `basil`, and finally to server `sage`.

Logging and Tracing of NFS Services

This section tells you how to start the following tools:

- NFS Logging
- AutoFS Logging
- AutoFS Tracing
- Logging for the Other NFS Services
- NIS Logging
- Logging With nettl and netfmt
- Tracing With nettl and netfmt

NFS Logging

You can configure logging for the following NFS daemons:

- `rpc.mountd`
- `rpc.statd`
- `rpc.lockd`

Each message logged by these daemons can be identifed by the date, time, host name, process ID, and name of the daemon that generated the message. You can direct logging messages from all these NFS daemons to the same file.

To Control the Size of Log Files

Log files grow without bound, using up disk space. You might want to create a `cron` job to truncate your log files regularly. Following is an example `crontab` entry that empties the log file at 1:00 AM every Monday, Wednesday, and Friday:

```
0 1 * * 1,3,5 cat /dev/null > log_file
```

For more information, type `man 1M cron` or `man 1 crontab` at the HP-UX prompt.

To Start and Stop rpc.mountd Logging

1. Issue the following commands to kill the rpc.mountd process and restart it with logging turned on (*PID* is a process ID returned by the ps command):

   ```
   ps -ef | grep mountd
   kill PID
   /usr/sbin/rpc.mountd -l /var/adm/mountd.log
   ```

2. If you want rpc.mountd to log mount requests and mount failures as well as errors, add the -t2 option to the rpc.mountd command, as in the following example:

   ```
   /usr/sbin/rpc.mountd -l /var/adm/mountd.log -t2
   ```

3. To stop logging, kill rpc.mountd and restart it without the -l *logfile* and -t2 options.

If you do not specify the -l or-t option, rpc.mountd logs only errors to /var/adm/mountd.log. If this file does not exist, rpc.mountd creates it. rpc.mountd can share the same log file with the other NFS daemons.

For more information, type man 1M mountd at the HP-UX prompt.

To Start and Stop Detailed Logging of rpc.statd and rpc.lockd

To start detailed logging of rpc.statd and rpc.lockd while they are running, issue the following commands (*PID* is a process ID returned by the ps command):

```
/usr/bin/ps -ef | /usr/bin/grep rpc.statd
/usr/bin/kill -SIGUSR2 PID
/usr/bin/ps -ef | /usr/bin/grep rpc.lockd
/usr/bin/kill -SIGUSR2 PID
```

The SIGUSR2 signal sets the logging to level 3 (the most detailed level).

The logging for rpc.statd is appended to the file /var/adm/rpc.statd.log. The logging for rpc.lockd is appended to the file /var/adm/rpc.lockd.log.

To stop detailed logging of rpc.statd and rpc.lockd, issue the same commands listed above to send the SIGUSR2 signal to the processes. The SIGUSR2 signal is a toggle that turns logging on or off, depending on its current state.

For more information, type man 1M statd or man 1M lockd at the HP-UX prompt.

To Start and Stop Basic Logging of `rpc.statd` and `rpc.lockd`

To start basic logging of `rpc.statd` and `rpc.lockd` (just errors, warnings, startup, and shutdown), issue the following commands (*PID* is a process ID returned by the `ps` command):

```
ps -ef | grep lockd
kill PID
ps -ef | grep statd
kill PID
/usr/sbin/rpc.statd -l /var/adm/rpc.statd.log
/usr/sbin/rpc.lockd -l /var/adm/rpc.lockd.log
```

NOTE Always start `rpc.statd` before starting `rpc.lockd`.

To stop basic logging of `rpc.statd` and `rpc.lockd`, kill them and restart them without the `-l` *logfile* option.

The `rpc.statd` and `rpc.lockd` daemons can share the same log file with the other NFS daemons.

For more information, type `man 1M lockd` or `man 1M statd` at the HP-UX prompt.

AutoFS Logging

The `automount` and `automountd` processes log messages through `/usr/sbin/syslogd`. By default, `syslogd` writes messages to the file `/var/adm/syslog/syslog.log`. Type `man 1M syslogd` for more information on `syslogd`.

To Enable `automount` Logging

The `automount` process runs at startup to parse the automounter maps and set up AutoFS mount points. It is not a daemon and does not run continuously. To enable `automount` logging, specify the `-v` option to the `automount` command, as follows:

```
/usr/sbin/automount -v
```

The `-v` option to `automount` causes it to log AutoFS mounts, unmounts, and other non-essential information to the console and to `syslog`.

To Start `automountd` Logging

1. Log in as root to the NFS client.
2. Issue the following command to stop `automountd`:

   ```
   /sbin/init.d/autofs stop
   ```

   If any automounted directories are currently in use, the `autofs` script will not stop `automountd` and will display a message.
3. If the `autofs` script failed to stop `automountd` because mounted directories were busy, warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can issue the following command to kill all the processes using the mounted directory:

   ```
   /usr/sbin/fuser -ck local_mount_point
   ```
4. In the `/etc/rc.config.d/nfsconf` file, use a text editor to add the `-v` option to the `AUTOMOUNTD_OPTIONS` variable, as follows:

   ```
   AUTOMOUNTD_OPTIONS="-v"
   ```

   The `-v` option causes `automountd` to log status messages to the console and to `syslog`.
5. Issue the following command to start `automountd` with logging enabled:

   ```
   /sbin/init.d/autofs start
   ```

To Stop automountd Logging

1. Log in as root to the NFS client.
2. Issue the following command to stop automountd:

   ```
   /sbin/init.d/autofs stop
   ```

   If any automounted directories are currently in use, the autofs script will not stop automountd and will display a message.
3. If the autofs script failed to stop automountd because mounted directories were busy, warn any users to cd out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can issue the following command to kill all the processes using the mounted directory:

   ```
   /usr/sbin/fuser -ck local_mount_point
   ```
4. In the /etc/rc.config.d/nfsconf file, use a text editor to remove the -v option from the AUTOMOUNTD_OPTIONS variable, as follows:

   ```
   AUTOMOUNTD_OPTIONS=""
   ```
5. Issue the following command to restart automountd with logging disabled:

   ```
   /sbin/init.d/autofs start
   ```

AutoFS Tracing

Two levels of Autofs tracing are available:

Detaled (level 3) Includes traces of all automounter requests and replies, mount attempts, timeouts, and unmount attempts. You can start level 3 tracing while `automountd` is running.

Basic (level 1) Includes traces of all automounter requests and replies. You must restart `automountd` to start level 1 tracing.

To Start and Stop `automountd` Detailed Tracing

1. Log in as root to the NFS client.
2. Issue the following commands (*PID* is the process ID returned by the `ps` command):

```
ps -ef | grep automountd
kill -SIGUSR2 PID
```

Level 3 tracing is sent to the console.

To stop level 3 tracing, issue the same commands listed above to send the `SIGUSR2` signal to `automountd`. The `SIGUSR2` signal is a toggle that turns tracing on or off depending on its current state.

If you have basic (level 1) tracing turned on when you send the `SIGUSR2` signal to `automountd`, the `SIGUSR2` signal turns tracing off.

To Start `automountd` Basic Tracing

1. Log in as root to the NFS client.
2. Issue the following command to stop `automountd`:

   ```
   /sbin/init.d/autofs stop
   ```

   If any automounted directories are currently in use, the `autofs` script will not stop `automountd` and will display a message.
3. If the `autofs` script failed to stop `automountd` because mounted directories were busy, warn any users to `cd` out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can issue the following command to kill all the processes using the mounted directory:

   ```
   /usr/sbin/fuser -ck local_mount_point
   ```

4. In the /etc/rc.config.d/nfsconf file, use a text editor to add the -T option to the AUTOMOUNTD_OPTIONS variable, as follows:

   ```
   AUTOMOUNTD_OPTIONS="-T"
   ```

   The -T option causes automountd to display each RPC call on standard output and log it to syslog.

5. Issue the following command to start automountd with tracing enabled:

   ```
   /sbin/init.d/autofs start
   ```

To Stop automound Basic Tracing

1. Log in as root to the NFS client.
2. Issue the following command to stop automountd:

   ```
   /sbin/init.d/autofs stop
   ```

   If any automounted directories are currently in use, the autofs script will not stop automountd and will display a message.

3. If the autofs script failed to stop automountd because mounted directories were busy, warn any users to cd out of the directory, and kill any processes that are using the directory, or wait until the processes terminate. You can issue the following command to kill all the processes using the mounted directory:

   ```
   /usr/sbin/fuser -ck local_mount_point
   ```

4. In the /etc/rc.config.d/nfsconf file, use a text editor to remove the -T option from the AUTOMOUNTD_OPTIONS variable, as follows:

   ```
   AUTOMOUNTD_OPTIONS=""
   ```

5. Issue the following command to restart automountd with tracing disabled:

   ```
   /sbin/init.d/autofs start
   ```

Logging for the Other NFS Services

You can configure logging for the following NFS services:

- `rpc.rexd`
- `rpc.rstatd`
- `rpc.rusersd`
- `rpc.rwalld`
- `rpc.sprayd`

Logging is not available for the `rpc.quotad` daemon.

Each message logged by these daemons can be identifed by the date, time, host name, process ID, and name of the function that generated the message. You can direct logging messages from all these NFS services to the same file.

To Control the Size of Log Files

Log files grow without bound, using up disk space. You might want to create a `cron` job to truncate your log files regularly. Following is an example `crontab` entry that empties the log file at 1:00 AM every Monday, Wednesday, and Friday:

```
0 1 * * 1,3,5 cat /dev/null > log_file
```

For more information, type `man 1M cron` or `man 1 crontab` at the HP-UX prompt.

To Configure Logging for the Other NFS Services

1. Add the `-l` *logfile* option to the lines in `/etc/inetd.conf` for the services you want to log. In the following example, logging is turned on for `rpc.rexd` and `rpc.rstatd`:

   ```
   rpc stream tcp nowait root /usr/sbin/rpc.rexd 100017 1 \
   rpc.rexd -l /var/adm/rpc.log

   rpc dgram udp wait root /usr/lib/netsvc/rstat/rpc.rstatd \
   100001 1-3 rpc.rstatd -l /var/adm/rpc.log
   ```

2. Issue the following command to restart `inetd`:

   ```
   /usr/sbin/inetd -c
   ```

If you do not specify a log file for the other NFS services (with the `-l` option), they do not log any messages. The NFS services can all share the same log file.

Type `man 1M rexd` for descriptions of the messages logged by the `rpc.rexd` daemon.

For more information, see the following man pages: `rexd`(1M), `rstatd`(1M), `rusersd`(1M), `rwalld`(1M), and `sprayd`(1M).

NIS Logging

You can configure logging for the following NIS processes:

- `ypxfr`
- `ypserv`
- `ypbind`
- `yppasswdd`

Each message logged by these daemons can be identifed by the date, time, host name, process ID, and name of the function that generated the message. You can direct logging messages from all these NIS daemons to the same file.

To Control the Size of Log Files

Log files grow without bound, using up disk space. You might want to create a `cron` job to truncate your log files regularly. Following is an example `crontab` entry that empties the log file at 1:00 AM every Monday, Wednesday, and Friday:

```
0 1 * * 1,3,5 cat /dev/null > log_file
```

For more information, type `man 1M cron` or `man 1 crontab` at the HP-UX prompt.

To Stop and Start Logging of `ypxfr`

If `ypxfr` is run interactively from the command line, it logs messages to standard output. If `ypxfr` is run by `cron` or by `yppush`, it logs messages to the file `/var/yp/ypxfr.log`, if the file exists. To start logging of `ypxfr`, issue the following command to make sure the `/var/yp/ypxfr.log` file exists:

```
/usr/bin/touch /var/yp/ypxfr.log
```

To stop logging of `ypxfr`, remove the `ypxfr.log` file:

```
/usr/bin/rm /var/yp/ypxfr.log
```

You cannot redirect the logging output of `ypxfr`.

For more information, see the following man pages: `ypxfr`(1M), `cron`(1M), and `yppush`(1M).

To Start and Stop Logging of ypserv

By default, the ypserv daemon logs messages to the file /var/yp/ypserv.log, if it exists. To start logging of ypserv, issue the following command to make sure the /var/yp/ypserv.log file exists:

```
/usr/bin/touch /var/yp/ypserv.log
```

To stop logging of ypserv, remove the ypserv.log file:

```
/usr/bin/rm /var/yp/ypserv.log
```

If you want to direct ypserv logging to a different file, follow these steps:

1. Add the -l *logfile* option to the YPSERV_OPTIONS variable in /etc/rc.config.d/namesvrs, as in the following example:

   ```
   YPSERV_OPTIONS="-l /var/yp/nis_log"
   ```

2. Issue the following commands to restart ypserv (*PID* is the process ID returned by the ps command):

   ```
   ps -ef | grep ypserv
   kill PID
   /usr/lib/netsvc/yp/ypserv options
   ```

   options is the list of options configured in the YPSERV_OPTIONS variable in the /etc/rc.config.d/namesvrs file. You can also source the /etc/rc.config.d/namesvrs file, and then enter the ypserv command as follows:

   ```
   /usr/lib/netsvc/yp/ypserv $YPSERV_OPTIONS
   ```

If you specify a log file with the -l option, ypserv can share the same log file with the other NIS daemons.

For more information, type man 1M ypserv at the HP-UX prompt.

To Configure ypbind Logging

1. Add the -l *logfile* option to the YPBIND_OPTIONS variable in /etc/rc.config.d/namesvrs, as in the following example:

   ```
   YPBIND_OPTIONS="-l /var/yp/nis_log"
   ```

2. Issue the following commands to restart ypbind (*PID* is the process ID returned by the ps command):

   ```
   ps -ef | grep ypbind
   kill PID
   /usr/lib/netsvc/yp/ypbind options
   ```

options is the list of options configured in the YPBIND_OPTIONS variable in the /etc/rc.config.d/namesvrs file. You can also source the /etc/rc.config.d/namesvrs file, and then enter the ypbind command as follows:

```
/usr/lib/netsvc/yp/ypbind $YPBIND_OPTIONS
```

If you do not specify a log file for ypbind (with the -l option), it logs messages to the system console, /dev/console. The ypbind daemon can share the same log file with the other NIS daemons.

For more information, type man 1M ypbind at the HP-UX prompt.

To Configure yppasswdd Logging

1. Add the -l *logfile* option to the YPPASSWDD_OPTIONS variable in /etc/rc.config.d/namesvrs, as in the following example:

   ```
   YPPASSWDD_OPTIONS="-l /var/yp/nis_log"
   ```

2. Issue the following commands to restart yppasswdd (*PID* is the process ID returned by the ps command):

   ```
   ps -ef | grep yppasswdd
   kill PID
   /usr/lib/netsvc/yp/rpc.yppasswdd options
   ```

   options is the list of options configured in the YPPASSWDD_OPTIONS variable in the /etc/rc.config.d/namesvrs file. You can also source the /etc/rc.config.d/namesvrs file, and then enter the yppasswdd command as follows:

   ```
   /usr/lib/netsvc/yp/rpc.yppasswdd $YPPASSWDD_OPTIONS
   ```

For more information, type man 1M yppasswdd at the HP-UX prompt.

Logging With `nettl` and `netfmt`

1. Issue the following command to make sure `nettl` is running:

   ```
   /usr/bin/ps -ef | grep nettl
   ```

   If `nettl` is not running, issue the following command to start it:

   ```
   /usr/sbin/nettl -start
   ```

2. Issue the following command to start logging:

   ```
   /usr/sbin/nettl -l i w e d -e all
   ```

   The logging classes are specified following the `-l` option. They are `i` (informational), `w` (warning), `e` (error), and `d` (disaster). Disaster logging is always on. You cannot turn it off. Information logging (`i`) fills up your log file faster than the other classes, so you might want to leave it off.

3. Recreate the event you want to log.

4. Issue the following command to turn logging off:

   ```
   /usr/sbin/nettl -l d -e all
   ```

   This command changes the logging class back to disaster only for all subsystems.

5. Issue the following command to format the binary log file:

   ```
   /usr/sbin/netfmt -lN -f /var/adm/nettl.LOG00 > formatted_file
   ```

   where *formatted_file* is the name of the file where you want the formatted output from `netfmt`. The default log file, `/var/adm/nettl.LOG`*nn*, is specified in the `nettl` configuration file, `/etc/nettlgen.conf`. If the file `/var/adm/nettl.LOG00` does not exist on your system, the default log file may have been changed in `/etc/nettlgen.conf`.

For more information, type `man 1M nettl` or `man 1M netfmt`.

Tracing With `nettl` and `netfmt`

1. Issue the following command to make sure `nettl` is running:

   ```
   /usr/bin/ps -ef | grep nettl
   ```

   If `nettl` is not running, issue the following command to start it:

   ```
   /usr/sbin/nettl -start
   ```

2. Issue the following command to start tracing:

   ```
   /usr/sbin/nettl -tn pduin pduout loopback -e all -s 1024 \
   -f tracefile
   ```

3. Recreate the event you want to trace.

4. Issue the following command to turn tracing off:

   ```
   /usr/sbin/nettl -tf -e all
   ```

5. Create the following filter file for `netfmt`:

   ```
   filter ip_saddr remote_host_IP_address
   filter ip_daddr remote_host_IP_address
   filter rpcprogram nfs
   filter rpcprogram nlockmgr
   filter rpcprogram llockmgr
   filter rpcprogram status
   filter rpcprogram mount
   filter rpcprogram portmap
   ```

   remote_host_IP_address is the IP address of the host with which your host was communicating when the event you want to trace occurred.

6. Issue the following command to format the binary trace file:

   ```
   /usr/sbin/netfmt -c filter_file -lN -f tracefile.TRC0 > formatted_file
   ```

   where *tracefile* is the name of the file you specified when you started tracing, and *formatted_file* is the name of the file where you want the formatted output from `netfmt`.

For more information, type `man 1M nettl` or `man 1M netfmt`.

Normal System Startup

This section explains the system startup sequence and how the NFS and NIS daemons are started up in a normal system boot.

1. The `/sbin/rc` script sources all the files in the `/etc/rc.config.d` directory. The files in `/etc/rc.config.d` contain environment variables that control the startup and behavior of various processes.
2. The `/sbin/rc` script runs the scripts in the directories `/sbin/rc0.d`, `/sbin/rc1.d`, `/sbin/rc2.d`, `/sbin/rc3.d`, and `/sbin/rc4.d`, in that order.

   The scripts in the `/sbin/rc`*n*`.d` directories are named S*NNNscriptname*, where *NNN* is a sequence number, and *scriptname* is the name of a startup script in the `/sbin/init.d` directory. Each of these scripts is actually a link to a startup script in `/sbin/init.d`. The `/sbin/rc` script runs them in order by sequence number. Following is a partial listing of the `/sbin/rc2.d` directory:

```
lrwxr-xr-x   1 root ... S400nfs.core -> /sbin/init.d/nfs.core
lrwxr-xr-x   1 root ... S410nis.server -> /sbin/init.d/nis.server
lrwxr-xr-x   1 root ... S420nis.client -> /sbin/init.d/nis.client
lrwxr-xr-x   1 root ... S430nfs.client -> /sbin/init.d/nfs.client
```

   All the startup scripts for the NFS services are started at run level 2 except the `nfs.server` script, which is started at run level 3. Table shows the NFS startup scripts, in the order they are run at system startup. It lists the processes that each script starts and the files and environment variables in `/etc/rc.config.d` that influence their behavior.

   All of the startup scripts start `portmap` if it is not already started, but only one `portmap` process should be running at once.

Startup Scripts for the NFS Services

Startup script in /sbin/init.d	Processes started	Related file in /etc/rc.config.d	Environment variables used
nfs.core	portmap(1M)	none	none
nis.server	portmap(1M) domainname(1) ypserv(1M) ypxfrd(1M) yppasswdd(1M) ypupdated(1M) keyserv(1M)	namesvrs	NIS_MASTER_SERVER NIS_SLAVE_SERVER NIS_DOMAIN YPSERV_OPTIONS YPPASSWDD_OPTIONS KEYSERV_OPTIONS YPUPDATED_OPTIONS YPXFRD_OPTIONS
nis.client	portmap(1M) domainname(1) ypbind(1M) keyserv(1M)	namesvrs	NIS_CLIENT NIS_DOMAIN WAIT_FOR_NIS_SERVER MAX_NISCHECKS YPBIND_OPTIONS KEYSERV_OPTIONS YPSET_ADDR
nfs.client	portmap(1M) biod(1M) statd(1M) lockd(1M) automount(1M) automountd(1M) mount(1M) swapon(1M)	nfsconf	NFS_CLIENT NUM_NFSIOD STATD_OPTIONS LOCKD_OPTIONS AUTOMOUNT AUTO_MASTER AUTOMOUNT_OPTIONS AUTOMOUNTD_OPTIONS
nfs.server	portmap(1M) exportfs(1M) mountd(1M) nfsd(1M) statd(1M) lockd(1M) pcnfsd(1M) swapon(1M)	nfsconf	NFS_SERVER NUM_NFSD STATD_OPTIONS LOCKD_OPTIONS START_MOUNTD MOUNTD_OPTIONS PCNFS_SERVER

Index

Index

Index

Index

Index

Index

Index

Index