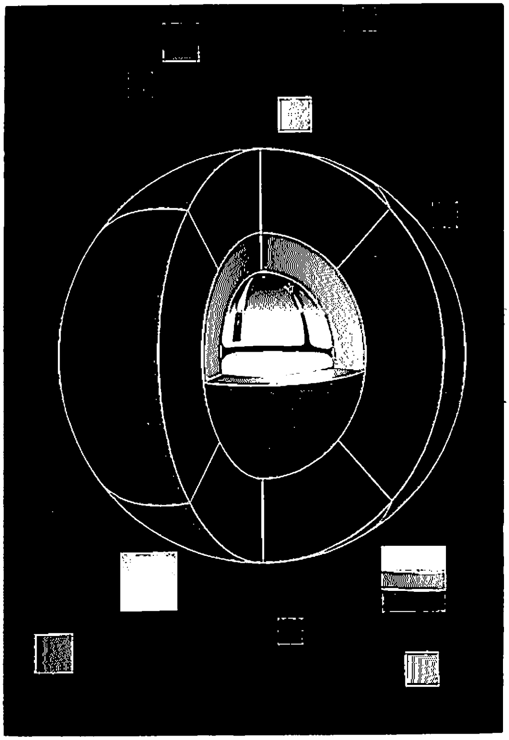
NCR System 3000
NCR UNIX SVR4 MP–RAS



An AT&T Company

Administrator Guide: Command Line Interface Devices and Networks (Volume 2) Release 2.02

September 1993

D1–2266–D

The program product(s) described in this book is a licensed product of NCR Corporation.

UNIX is a registered trademark of UNIX Systems Laboratories.

It is the policy of NCR Corporation to improve products as new technology, components, software, and firmware become available. NCR Corporation, therefore, reserves the right to change specifications without prior notice.

All features, functions, and operations described herein may not be marketed by NCR in all parts of the world. In some instances, photographs are of equipment prototypes. Therefore, before using this document, consult your NCR representative or NCR office for information that is applicable and current.

To maintain the quality of our publications, we need your comments on the accuracy, clarity, organization, and value of this book. Address correspondence to:

Information Products
NCR Corporation
3325 Platt Springs Road
West Columbia, SC 29170
U.S.A.

Preface

Who Should Read This Book

This book is for the knowledgeable system administrator, that is, the person who administers the system using the operating system commands at the shell prompt (# for the **root** login).

Releases Covered In This Book

This book applies to Release 2.02 of the operating system for your NCR System 3000 computer.

What You Should Know

You should be familiar with the information in the following books:

- *User Guide: Introduction to NCR UNIX SVR4*
- *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface*

How To Use This Book

This book assumes you are familiar with the *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface* and takes you one step beyond that book to include:

- Command line procedures that perform some of the same tasks that the menu procedures perform in the *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface*

- Command line procedures for many more advanced tasks
- Diagnostics and trouble–shooting procedures

Rather than lead you through chapter by chapter, this book provides a reference in which you can look up a particular topic or task. Use the *NCR UNIX SVR4 MP–RAS Reference Manual* as needed to obtain additional information about a particular command.

This book consists of the following volumes:

Volume 1 — General Administration
This volume describes the following:

- System administration logins, duties, and schedules
- Changing run levels
- Using the Reference, Diagnostics, and boot flex diskettes
- Recovery procedures using the Maintenance (*/install*) file system
- Administering, checking, and repairing the file system
- Backing up and restoring information

Volume 2 — Devices and Networks
This volume describes the following:

- Installing additional disks, adapters, and terminals
- Setting up and managing the LP Print Service
- Setting up and administering the Basic Networking Utilities (BNU)

Volume 3 — System Configuration

This volume describes the following:

- The files and directories used in configuration
- The process of reconfiguring the system and building new kernels
- The tunable configuration parameters

Appendices and Glossary

The appendices describe the file system, device names and numbers, customizing the sysadm interface, using national characters on the console, and using sendmail. The glossary defines terms you encounter in this book.

Conventions Used In This Book

- The following type identifies text that you must enter exactly as shown:

  `login su`

- The following type identifies output from a screen or command:

  `Press the Return key to continue.`

- Path names and file names appear in italics. For example:

  The */etc/profile* file defines the standard environment for all users.

- Configuration parameters and shell variables appear in capital letters. Capital letters are used for emphasis also. For example:

  Press RETURN after specifying a value for the TERM variable. Do NOT leave this value blank.

- Utilities, commands, user names, and terminal keys appear in boldface type. For example:

  The **cpio**(1) command backs up files.

- Parameters for command line options appear in italics within the text. For example:

  Specify the class (–c*class*) for the printer.

- The numbers and letters in parentheses after a command name indicate the type of command.

  (1)
  Standard command available to users at the $ prompt.

  (1M)
  Command available only to someone logged in as **root**

  (2)
  System call

  (3)
  Library function

Related Publications

For supplementary information, refer to the following books:

- *User Guide: Introduction to NCR UNIX SVR4*
- *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface*
- *NCR UNIX SVR4 Message Manual*
- *NCR 345x/35xx Unit Installation, Care, and Cleaning*
- *NCR UNIX SVR4 MP–RAS Reference Manual*

Contents

Volume 1 — General Administration

Chapter 1

Common System Administration Tasks

Chapter 2

Using the Maintenance and Reference Diskettes

Chapter 3

Recovery Techniques

Chapter 4

Administering the File System

Chapter 7

Restoring Information

Chapter 8

Administering the UNIX SVR4 Mail Subsystem

Appendix A

The File System

Appendix B

Device Names and Numbers

Appendix C

Customizing the sysadm Interface

Appendix D

Using National Characters on the Console

Appendix E

Sendmail – An Internetwork Mail Router

Appendix F

Sendmail Installation and Operation

Volume 2 — Devices and Networks

Chapter 1

Configuring Disks

Chapter 2

Configuring Adapters

Chapter 3

Configuring Serial Devices

Chapter 4

Using The LP Print Service

Chapter 9

Managing BNU

Volume 3 — System Configuration

Chapter 1

Monitoring System Activity

Chapter 2

Process Scheduling

Chapter 3

Configuring Kernels

Chapter 4

Configuration Parameters

Chapter 5

Tuning Basics

Chapter 6

Memory Utilization

Chapter 7

Disk Utilization

Chapter 8

CPU Utilization

Figures

Volume 1 — General Administration

Volume 2 — Devices and Networks

Volume 3 — System Configuration

Chapter 1

Configuring Disks

This chapter contains the procedures for the following:

NOTE: It is strongly recommended that you use the **sysadm** menus to perform the tasks in this chapter. Use these procedures ONLY if you are unable to use the menus.

If you use these procedures, it is very important that you use the **mktable** command when you are finished so the information in the object management table (*/etc/device.tab)* will be updated to match your actual configuration.

Using Disk Utilities

The following utilities are available for managing disks:

dkformat(1M)
Formats a fixed disk.

format(1)
Formats a flex diskette.

fdisk(1M)
Partitions the disk and creates the master boot block.

dklayout(1M)
Initializes the UNIX partition (initializes the boot code and creates the pdinfo and vtoc structures).

swap (1M)
Adds swap space (if the disk contains a slice to be used as swap space).

fdump(1M)
Adds dump space (if the disk contains a slice to be used as dump space.

edvtoc(1M)
Allows you to edit the contents of the Volume Table of Contents.

prtvtoc(1M)
Displays the contents of the Volume Table of Contents, including the slice number, slice tag, slice flag/permissions, slice start sector, and slice size.

Adding Disks

Why/When

You may add a disk your system to:

- Improve system performance by distributing file systems across disks.
- Provide additional storage space.

Additional Information

It is strongly recommended that you use the menus to perform this task. If you use this procedure instead of the menus, be sure to run **mktable** when you are finished.

Important!

It is always a goot idea to back up any important data before reconfiguring a system.

Before You Start

You should do the following before adding a disk:

- Make sure no users attempt to access the disk.
- Make sure the disk is physically added to the system and, if necessary, strapped with the appropriate unique SCSI ID.
- If the disk is contained in an external unit, make sure the power to that unit is ON.

- Verify that the block and character special device files for slice 0 are in the */dev* directory. Check for these files by entering:

```
# ls -l /dev/rdsk/* | pg
```

NOTE: If the devcie node does not exist, use the **nodes**(1) utility to determine the major/minor numbers and the node name.

Procedure

The details for the steps of this procedure are in the following sections of this chapter.

Step 1: Log in as **root**.

Step 2: Format the disk (if it is not already formatted).

NOTE: If you are adding an internal disk array, use the RAID Manager to format the disk. See the *NCR RAID Manager 3.0 User Guide for UNIX V.4* for additional information.

Step 3: Partition the disk.

Step 4: Write boot code and initialize the disk.

Step 5: Slice the disk.

Step 6: Create file systems, if desired.

Step 7: To make sure the slices are set up properly, do the following:

- Examine the contents of the */etc/vfstab* file. An entry should appear for each file system you have added.
- Perform the **mount –v** command to determine if the file systems you intend to be automatically mounted are mounted.
- Perform the **df –g** command to display information about the file systems.

Step 8: Update the device table.

References

Refer the *Reference Manual* for information about **dkformat(1M)**, **fdisk(1M)**, **dklayout(1M)**, **edvtoc(1M)**, **prtvtoc(1M)**, **fdump(1M)**, and **swap(1M)**.

Formatting Disks

Why/When

A disk must be formatted before your system can use it.

In most new systems, disks are formatted before shipment. If not formatted previously, the root device is formatted automatically during installation. Some reasons you may need to format a disk include:

- To add a new disk device to your system
- To add new bad block information (a nondestructive, maintenance format). See the section "Marking Bad Blocks" in this chapter.
- To recreate your file system (for example, if the integrity of your file system is destroyed)

Before You Start

Make sure the disk is physically added to the system and strapped correctly.

Back up any file systems that contain information you want to keep.

Caution

Formatting destroys all information on the disk.

Procedures

To format a flex diskette:

Step 1: Place the diskette in the flex drive.

Step 2: Enter the **format**(1) command.

For example, to format a high density/double sided diskette, enter:

```
# format /dev/rdsk/f03ht
```

To format a hard disk:

Step 1: If the disk is external, be sure the power to the unit on.

Step 2: Enter the **dkformat**(1M) command:

```
# dkformat -d /dev/rdsk/device_name
```

The *device_name* is the name of the disk you want to format. See Appendix B for an explanation of device names.

Partitioning Disks

Why/When

You must partition a disk before it can be used to store the operating system(s) for your computer. A disk may contain one partition (for 100% NCR UNIX), or it may contain multiple partitions (for NCR UNIX, DOS, and others).

In addition, you may want to repartition an existing disk to add or remove DOS or other non–UNIX partitions.

Additional Information

When you partition a disk, you are asked to specify the "active" partition. This is the partition that is used when the system boots. To change active partitions (for example, to boot DOS instead of NCR UNIX), see the section "Changing the Active Partition."

Figure 1–1 illustrates a disk that contains a single 100% NCR UNIX partition. Figure 1–2 illustrates a disk containing multiple partitions (UNIX, DOS, etc.).

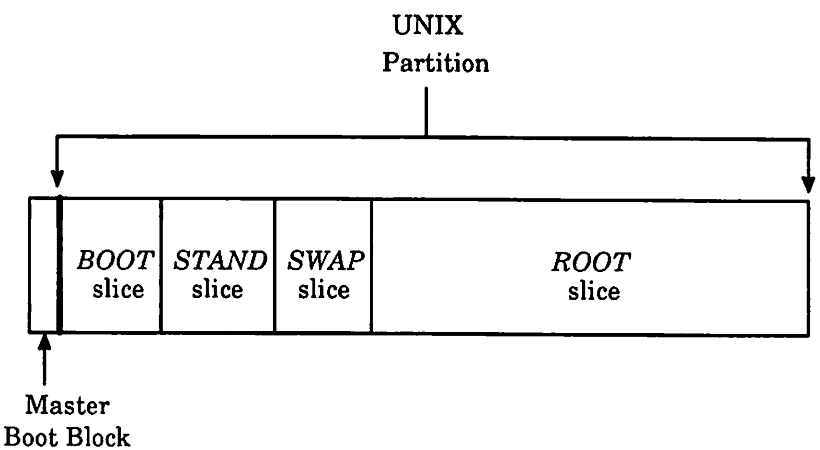


Figure 1–1: A Disk Containing a 100% NCR UNIX Partition

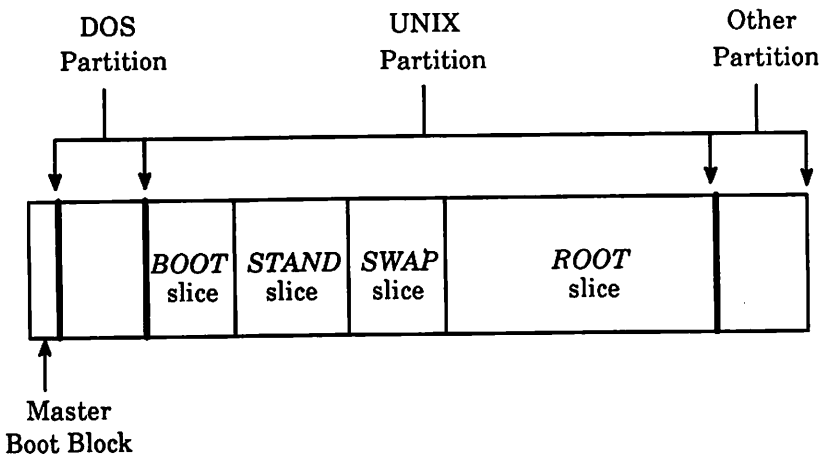


Figure 1–2: A Disk Containing Multiple Partitions

Restrictions

There are certain restrictions on the combinations and placement of partitions, depending on the operating systems you plan to install on the partitions.

- If you have a DOS partition on the first disk, it must be in the first partition and the DOS operating system should be installed before you install NCR UNIX.
- If you put a DOS partition on a second disk, you must boot DOS from the flex and then change to the C directory on the second disk to use DOS.

Before You Start

- Back up any data you want to keep before you partition a disk that is already in use on your system.
- Make sure the disk is strapped correctly, if necessary, and physically added to the system.

Procedure

This procedure involves formatting the disk as well as creating partitions on it.

Caution

Partitioning destroys all information that exists on the disk.

NOTE: It is best to perform this procedure on the console. If you must use the X Window System, set the window size to 80x24 and set the TERM and LINE environment variables as follows:

```
# TERM=xterm
# LINES=24
# export TERM LINES
```

Step 1: Make sure the disk is formatted.

Step 2: Partition the disk with the **fdisk**(1M) command:

```
# fdisk /dev/rdsk/device_name
```

The *device_name* is the name of the device you are partitioning. See Appendix B for an explanation of device names.

A screen similar to Figure 1–3 appears:

```
Total hard disk size is 775 cylinders
                                                      Cylinders
Partition   Status     Type         Start   End    Length   %
=======     =====      ====         ====    ====   ======   ==
   1        Active   UNIX System     0      774     775    100

SELECT ONE OF THE FOLLOWING:

          1. Create a partition
          2. Change Active (Boot from) partition
          3. Delete a partition
          4. Exit (Update a disk configuration and exit)
          5. Cancel (Exit without updating disk configuration)

Enter Selection:
```

Figure 1–3: The fdisk Utility Screen

The number of cylinders that appears on this screen differs according to the size of your disk. The disk in Figure 1–3 is 100 MB. When editing this screen, you can specify partition size by number of cylinders or by percentage of disk.

Step 3: Enter the number of the change you want to make.

The **fdisk** utility allows you to perform the following tasks:

- Create a partition, specifying the type (for example, UNIX, DOS), amount of disk it should occupy, and whether it is the active partition.
- Change the active partition, that is, the partition the system boots from. Note that you must specify an active partition for each disk on the system in order to access that disk.

- Delete a partition so that you can repartition the disk.

- Update the disk according to the changes that you have made and exit the **fdisk** utility.

- Cancel any changes you have made and exit the **fdisk** utility without modifying the disk.

Step 4: When the disk is partitioned the way you want, enter Selection 4, Exit.

Step 5: If you do NOT plan to slice the disk right away, run **mktable** to update the object table:

```
# /usr/sadm/sysadm/bin/mktable
```

References

Refer to **fdisk**(1M) in the *Reference Manual* for more information about command syntax and usage.

Writing Boot Code and Initializing Disks

If you have formatted and partitioned the disk but have not written the boot code to the disk, use this command to write the boot code, initialize the physical device information, and initialize the volume table of contents (vtoc).

Enter the following command:

```
# dklayout -b -p -v -d /dev/rdsk/device_name
```

- **–b** writes boot code
- **–p** initializes physical device information
- **–v** initializes volume table of contents (vtoc)

Slicing Disks

Why/When

A slice is a subset of a UNIX partition and is treated as a logical device. Each UNIX partition should have slice 0 (a reference for accessing the entire partition) and at least one file system slice.

You may choose to create additional slices to:

- Limit the amount of disk space a certain user, data base, or user group may use
- Facilitate networking access
- Enhance system performance
- Enhance security for sensitive information

> **Caution**
>
> If you reduce the size of an existing slice, you may overwrite part of an existing file system, and lose the data stored there.

Before You Start

- Determine which slices you want to add to your system. Figure 1–4 indicates the appropriate use for each slice of your boot disk. (On non–boot disks, you can use slices 1 through 6 and 8 through 15 for file systems.)

Slice	Use
0	To access the entire UNIX partition
1	Default file system *(root* if the root disk)
2	The SWAP slice
3 – 4	For file systems
5	For a DOS partition (if present)
6	For file systems (default DUMP if root disk)
7	The BOOT slice (Do not change this even if the disk is not the BOOT device. Partitioning information as well as boot code resides here.)
8 – 9	For file systems
10	The STAND slice for the */stand* file system
11 – 15	For file systems

Figure 1–4: Appropriate Slice Assignments

- Log in as **root** to slice a disk.
- Put the system in single user mode and warn users to log off.
- Use the installation tape to slice the *root* disk. To change slicing of any system file systems (such as / and */stand*), you must re-install the operating system.
- Make sure you have a backup (**cpio**) of all file systems that reside on the hard disk being sliced.

Procedure

Step 1: Create an ASCII file that contains slicing information by entering:

```
# cd /
# prtvtoc -f slices /dev/rdsk/device_name
```

The –f option places the output of the **prtvtoc** command in the ASCII file *slices*. The *device_name* is the name of the disk you are slicing. See Appendix B for an explanation of device names.

Step 2: Use a system editor to change the values in the *START* and *SIZE* columns in the ASCII file.

To prevent data loss due to overlapping slices, make sure the starting block number of a slice is greater than the starting block number plus the size (length) of the preceding slice. The following illustration shows a possible file for a 760MB disk.

#SLICES	TAG	START	SIZE
0	0x5	1	1307900
1	0x4	307901	1000000
2	0x4	35	307866
3	0x0	0	0
4	0x0	0	0
5	0x0	0	0
6	0x0	0	0
7	0x1	1	34
8	0x0	0	0
9	0x0	0	0
10	0x0	0	0
11	0x0	0	0
12	0x0	0	0
13	0x0	0	0
14	0x0	0	0
15	0x0	0	0

The TAG column indicates the type of slice. Do NOT change the value in the TAG column for slices 0 and 7. The prinary boot device must also have a STAND slice and a SWAP slice, created during system installation. For other slices, you may change the TAG to one of the values in Figure 1–5, depending on how you plan to use the slice, but you are not required to do so:

Tag	Description	Value
BOOT	Boot slice	0x01
ROOT	Root file system	0x02
SWAP	Swap slice	0x03
USR	Usr file system	0x04
BACKUP	Full disk	0x05
ALTS	Alternate sector space	0x06
OTHER	Non-UNIX space	0x07
ALTTRK	Alternate track space	0x08
STAND	Stand file system	0x09
VAR	Var file system	0x0a
HOME	Home file system	0x0b
DUMP	Dump slice	0x0c
RESERVED	Reserved raw slice	0x0d

Figure 1–5: TAG Names and Descriptions

Step 3: Enter the following command to update the slice information on the hard disk:

```
# edvtoc -f slices /dev/rdsk/device_name
```

The *device_name* is the name of the device you are slicing. See Appendix B for an explanation of device names.

Your screen displays the new slicing information followed by the message:

```
Enter 'y' to select the above slices, 'n' to
quit
```

Step 4: Verify that the slicing information is correct.

- If the slicing information is correct, enter the letter **y**.
- If the slicing information is NOT correct, enter the letter **n** and repeat Steps 2 and 3.

Step 5: Use the **mkfs**(1M) command to create the desired file systems in each slice.

Step 6: Restore all file systems that were backed up in Step 1.

Step 7: Verify that all files are restored correctly.

Step 8: Put the system in multiuser mode (run level 2).

Step 9: If you do not plan to make file systems on the disk right away, run **mktable** to update the object table:

```
# /usr/sadm/sysadm/bin/mktable &
```

Marking Bad Blocks

Why/When

Marking bad blocks ensures that the operating system does not attempt to write data to these areas of your disk.

Before You Start

- Determine whether bad (defective) blocks have been found on your system; use the **/etc/errpts/SCSI_Errpt** command and examine the contents of the system error log file(s) in the */var/adm/streams* directory.

  If a bad block was found by a SCSI controller, the following line should be displayed in one or more error records:

  ```
  ***** BAD BLOCK FOUND AT 0xyyyyyyyy
  ```

  The value 0x*yyyyyyyy* indicates the logical block number, in hexadecimal, of the bad block.

- Ensure that users are not logged into the system and that daemon processes are not running. Use one of the following methods:

  - Put the system in single user mode

    OR

  - Use the maintenance file system to perform this procedure

Procedure

In this procedure, examples use the bad block number 0x000141A8.

Step 1: Log in as **root**.

Step 2: Enter the **dkformat** command to mark defective/bad block(s).

For example, enter :

```
# dkformat -d /dev/rdsk/device_name -v -m -u -
```

You must use the character special device name and you must specify slice 0.

Options used in this example are as follows:

–v (verbose) indicates that you are permitted to correct any input errors while performing the **dkformat** utility.

–m indicates that a full format is disabled (maintenance only).

–u – indicates that you will enter the defective block numbers from a terminal.

Step 3: To continue with this procedure, enter the letter **y** (yes) after the following prompt is displayed:

```
Enter 'y' to continue, 'n' to quit:
```

To exit the **dkformat** utility, enter the letter **n** (no).

Step 4: Enter the hexadecimal number, or the decimal number, for the bad block at the following prompt:

```
Defect #1  (^D or $ to quit):
```

If you enter the hexadecimal number, you must enter the **0x** (prefix) to indicate hexadecimal but you do not have to enter the leading zeroes (0). For example, you can enter **0x141A8.**

If you want to enter the decimal number, convert the hexadecimal number to the corresponding decimal number. Then, enter that decimal number, such as **82344**.

Step 5: Enter the appropriate response to the following prompt:

```
Block 82344 (0x141A8) correct?  (y/n)
```

If the block number is NOT correct, enter the letter **n** (no) and you are prompted to re–enter the defective block number (the prompt shown in step 4 is displayed again).

Step 6: Repeat steps 3, 4, and 5 for each bad block.

Step 7: After you enter all bad block numbers, exit the **dkformat** utility by entering CTRL–d or a dollar sign ($) character.

Step 8: To examine the field grown defect list, enter the following command:

```
# badlist  -d  /dev/rdsk/device_name
```

An example of the output is as follows:

```
311  7  5862
```

The first value is the cylinder number, the second value is the head number, and the third value is the byte offset.

Step 9: Unmount each file system that contained a bad block and then perform the **fsck** utility on that file system.

If the bad block was in the root file system, you can do this using the maintenance file system.

Step 10: If you are using the maintenance file system, enter CTRL–d to exit the maintenance file system, remove the second install flex disk from the flex disk drive, and then reboot the system.

Displaying Slicing Information

Why/When

Use the **prtvtoc**(1M) command to display slicing information about the hard disk or to store this information in a file. Non–UNIX partitions show up in this display, also.

Use the **edvtoc**(1M) command to edit slicing information about the hard disk.

Procedure

To display slicing information, use the **prtvtoc** command with no options. For example,

```
#    prtvtoc   /dev/rdsk/device_name
```

The *device_name* is the name of the disk for which you are displaying slicing information. See Appendix B for an explanation of device names.

With the **–p** option, the **prtvtoc** command displays the vendor and product ID for the specified device. For example,

```
#    prtvtoc   -p   /dev/rdsk/device_name
```

References

Refer to the *Reference Manual* for more information about **prtvtoc**(1M) and **edvtoc**(1M).

Changing the Active Partition

Why/When

Your system boots from the active partition. If you want to boot from a partition other than the one you are currently using, you must specify a different active partition.

Before You Start

Warn users to log off and put the system in single user mode.

Procedure

Step 1: Enter the following command:

```
# fdisk /dev/rdsk/device_name
```

Step 2: Enter **2** to select *Change Active Partition.*

Step 3: Enter the number of the partition you want to make active. (The possible partitions are displayed.)

Step 4: Reboot the system.

- If you are currently running UNIX, use the **shutdown** command. For example,

```
# shutdown -i6 -g0 -y
```

- If you are currently running DOS, press the ESC key twice to return to the DOS prompt, then reboot by pressing the CTRL–ALT–DEL keys.

Configuring Swap Space

Initial Swap Configuration

The amount of swap space your system needs is highly dependent on the system load. A system starts swapping when a process has used its time allocation or when the process can not complete because it is waiting for I/O. When you install the BASE operating system, swap is created as shown in Figure 1–6:

Memory Range	Swap Space initially installed
≥ 16 MB ≤ 128 MB	2 x memory size
> 128 MB ≤ 256 MB	256 MB
> 256 MB ≤ 512 MB	1 x memory size
> 512 MB	512 MB

Figure 1–6: Initially Installed Values for Swap Space

Setting up Additional Swap Space

If additional swap space is needed, it is recommended that you divide any additional swap space among available disk drives. When swap space is needed, swap space is allocated sequentially, but in a round robin type of distribution across all available swap sources.

NOTE: Use the numbers in Figure 1–6 as a guideline and use the utilities **sar –r** and **sar –w** to monitor free memory and swap space in order to make necessary reconfigurations for swap space allocation.

Caution

If you plan to rearrange the original swap space allocation setup by the installation, you must always leave some amount of primary swap space (*/dev/swap*) so that the system can reach the swapadd rc script to add additional swap. If you do not have enough primary swap space, the system will hang during the booting process.

Procedures to Add Swap Space

You can increase the amount of available swap space by enlarging your swap slice, creating a secondary swap slice on your disk, or creating a swap file. If you do not anticipate needing the increased swap space frequently, swapping to a file may suffice. However, it is more efficient to use a slice rather than a file for swapping.

Adding Swap Space to a Disk Slice

Step 1: Use the "Slicing Disks" procedure to create a swap slice, changing the tag of the chosen FREE area (0x00) to SWAP (0x03).

Step 2: Use the **swap(1M)** command to add swap space to your disk.

```
# swap -a device swaplow swaplen -u
```

- *device* names a SCSI device. See Appendix B for an explanation of device names.
- *swaplow* specifies the offset in 512 byte blocks from the beginning of the slice.
- *swaplen* specifies the length of swap area to be added in 512 byte blocks.

For example, to add 5 MB of additional swap space on slice 2 of a non–root disk with an offset of zero blocks into the slice, type the following command:

```
# swap -a /dev/dsk/c0t0d0s2 0 10240 -u
```

NOTE: The **–u** option will update the file */etc/swapnodes* which lists all the additional swap devices and/or swap files. If this option is omitted, the added swap will be lost upon the next reboot of the system. Thus, without this option you can temporarily add swap space.

Step 3: Run the **mktable** command to update the area type for the new swap area in the object table:

```
# /usr/sadm/sysadm/bin/mktable &
```

The following procedures shows how to create and remove a swap file. The name of the swap file in the examples is */swap2*.

Adding Swap Space to a Disk File

Step 1: Log in as **root** and find out the maximum size of a single file on your system by entering the following command:

```
# ulimit
```

This command will return a number of 512 byte blocks. If the value returned is smaller than the number of blocks you want to add for additional swap space, you must perform the next command:

```
# ulimit num-of-blocks
```

For *num–of–blocks* you must enter the number of physical blocks (512–bytes each) that are to be allocated to the swap file. For example, if you want the swap file to have a size of 5 MB, enter:

```
# ulimit 10240
```

Step 2: Create a file in any directory. For example, enter:

```
# dd if=/dev/zero of=/swap2 bs=512 \
 count=10240
```

Make sure you enter the same number of blocks in the *count* field that you entered in Step 2.

Step 3: Use the **chmod** command to change the permissions on the swap file:

```
#   chmod  600  /swap2
```

Step 4: Add the swap file to the swap area used by the kernel:

```
#   swap  -a  /swap2  0  10240 -u
```

NOTE: The **–u** option will update the file */etc/swapnodes* which lists all the additional swap devices and/or swap files. If this option is omitted, the added swap will be lost upon the next reboot of the system. Thus, without this option you can temporarily add swap space.

Step 5: To make sure the swap file is configured, enter:

```
#   swap  -l
```

Procedures to Remove Swap Space

If you find that the swapping activity has decreased on your system and you no longer need to swap to a slice or part of a slice, you should remove it. If you find that you no longer need to swap to a file, which was added to overcome an increased need for swapping, you should remove it. You can do so using the following procedures.

Removing Swap from a Slice

Step 1: Perform the following command to prevent the kernel from swapping to the file:

```
#   swap  -d  device swaplow -u
```

For example:

```
#   swap  -d  /dev/dsk/c0t0d0s2  0  -u
```

NOTE: The **–u** option will update the file */etc/swapnodes* which lists all the additionala swap devices and/or swap files. If this option is omitted, the entry for the deleted swap space will not be removed from */etc/swapnodes*. When the system reboots all swap devices and/or files as listed in */etc/swapnodes* will be added as swap space. Thus, if you don't specify this option, the swap space will only be removed temporarily.

Step 2: Perform the following command to ensure that the swap to the slice has been deallocated:

```
#   swap -l
```

Caution

If the slice is still reported and you see INDEL beside the name of the slice name, it is in the process of being deleted. Do NOT use the slice or the part of the slice you used for swapping for another purpose until the file is no longer being reported by the swap –l command.

Step 3: Run the **mktable** command to update the object table with the deleted swap space:

```
# /usr/sadm/sysadm/bin/mktable &
```

Removing a Swap File

Step 1: Perform the following command to prevent the kernel from swapping to the file:

```
#   swap  -d  name_of_swap_file swaplow -u
```

For example:

```
#   swap  -d  /swap2  0  -u
```

NOTE: The **–u** option will update the file */etc/swapnodes* which lists all the additional swap devices and/or swap files. If this option is omitted, the entry for the deleted swap space will not be removed from */etc/swapnodes*. When the system reboots all swap devices and/or files as listed in */etc/swapnodes* will be added as swap space. Thus, if you don't specify this option, the swap space will only be removed temporarily.

Step 2: Perform the following command to ensure that the swap to a file has been deleted:

```
#   swap -l
```

Caution

If the file is still reported and you see INDEL beside the name of the file name, it is in the process of being deleted. Do NOT proceed to step 3 until the file is no longer being reported by the **swap –l** command.

Step 3: Delete the swap file from the disk by entering:

```
#   rm  /swap2
```

Configuring the Dump Area

Why / When You Need a Dump Area

A memory dump is an image of the contents of RAM. Dumps are performed in the following instances:

- When the system crashes upon encountering a fatal error in the operating system, the contents of memory are written to hard disk.

  A systems analyst may then examine the dump to determine the operating system environment at the time of the failure in order to better understand what may have caused the system crash.

- When the system hangs, the system administrator can manually issue a dump via the dump switch on the NCR systems 3450 and 355x.

  A systems analyst may then examine the dump to determine the environment at the time of the system hang in order to better understand what may have caused the system to hang.

- When the system is setup for powerfail recovery and when power is lost, the system dumps the contents of memory to the dump area in order to be able to bring itself back to the point of execution once power is restored. If the dump area is not set up properly, the system will not be able to perform a powerfail recovery. Refer to Chapter 1, "Being a System Administrator" for more information about how to setup the system for powerfail recovery.

NOTE: The dump area must always be at least equal to the size of physical memory (see **dumpconfig**(1M)).

Tasks Involved for Dumps

Managing dumps requires the following tasks:

- Configuring the Dump Area (as described in this section)
- Customizing Memory Dumps (Refer to Chapter 1, "Being a System Administrator")
- Saving Memory to Dump Area (Refer to the "Recovery Techniques" chapter in Volume 1)
- Saving the Dump Area to Media (Refer to the "Recovery Techniques" chapter in Volume 1)
- Analyzing the Dump (Refer to the "Recovery Techniques" chapter in Volume 1)

Initial Dump Configuration

On systems with 512 MB of memory or less, dump is created equal in size to memory during installation of the BASE operating system and stored in Slice 6 of the root disk by default, as shown in Figure 1–7. For systems with more than 512 MB of memory, the installation process sets up only 512 MB of dump space since there would not be enough disk space to install the operating system software and the required swap space to initially boot the system. Since the dump space must be equal to the size of the memory, you are instructed to add more dump space after the initial installation.

Memory Range	Dump Space initially installed	Required Dump Space
≥ 16 MB ≤ 512 MB	1 x memory size	1 x memory size
> 512 MB ≤ 2048 MB	512 MB	1 x memory size

Figure 1–7: Initial and Required Values for Dump Space

Setting up Additional Dump Space

Dump space must always be equal to the size of memory. For systems with a large memory size, you should always distribute the dump space across different SCSI channels to enable parallel dumps. The dump space can be located on any disk(s). You can only have a maximum of 512 MB dump space per SCSI channel. With 2 GB memory, this means you must distribute the dump space across at least four SCSI channels.

NOTE: Dump space can not be put on a shared device.

Procedure to Add Dump Space

Step 1: Use the "Slicing Disks" procedure to create a dump slice, changing the tag of the chosen FREE area (0x00) to DUMP (0x0c).

Step 2: Use the **fdump**(1M) command to add dump space to your disk.

```
# fdump -a device dumplow dumplen
```

- *device* names a SCSI device. See Appendix B for an explanation of device names.
- *dumplow* specifies the offset in 512 byte blocks from the beginning of the slice.
- *dumplen* specifies the length of dump area to be added in 512 byte blocks.

For example, to add 5 MB of dump space on slice 6 of a non–root disk with an offset of zero blocks into the slice, type the following command:

```
# fdump -a /dev/dsk/c0t0d0s6 0 10240
```

Step 3: Run the **mktable** command to update the area type for the new dump area in the object table:

```
# /usr/sadm/sysadm/bin/mktable &
```

Validating Dump Area

You can validate the system dump area using the **dumpconfig**(1M) command. This command is very useful because it allows you to:

- Check if there is enough dump space allocated.
- Verify the SCSI bus distribution so that powerfail recovery is possible.
- Report the total number of disk blocks needed for successful dumping of memory.
- Report the number of disk blocks currently configured for a specified (primary or secondary) Micro Channel, SCSI controller and SCSI bus.

For example, use the following command line to compare the allocated dump space with the memory size:

```
# dumpconfig -m
```

If the command returns a **0**, then you have enough dump space allocated in your system.

Procedure to Remove Dump Space

Step 1: Perform the following command to remove dump space from your system:

```
#   fdump -d device swaplow
```

For example:

```
#   fdump  -d  /dev/dsk/c0t0d0s6  0
```

Step 2: Run the **mktable** command to update the object table with the deleted dump space:

```
# /usr/sadm/sysadm/bin/mktable &
```

Replacing a Failed Disk

On an NCR 3550, you can replaced a failed disk while the system remains in service in multiuser mode. You can replace any of the following:

- SCSI fixed disks
- SCSI removable media devices
- SA/400 removable media devices

Procedure

It is highly recommended that you use the Hardware Manager feature of the Open Systems Administrator to replace failed disks. If you are unable to use the Hardware Manager, perform these steps:

Step 1: Power down the device:

```
# cd /usr/sadm/sysadm/bin
# OTperform -A diskalias -M poweroffunc
```

The *diskalias* is the alias name used by the object table for the device. For example, the disk */dev/dsk/c0t6d0s0* has an alias of *c0t6d0s0.*

Step 2: Replace the device. Make sure you remove the device you turned off the power to in Step 1.

The new device should have at least the same capacity as the old disk. If the new device is smaller, you can not rebuild the information on the disk. If the new disk is larger, extra space may not be accessible once you rebuild the disk.

Step 3: Power up the device:

```
# OTperform -A  diskalias -M poweronfunc
```

The *diskalias* is the same diskalias you used in Step 1.

Step 4: Use the **sysadm** menus to format the new disk. If you can not use the menus, use **dkformat**(1M).

Step 5: Use the **sysadm** menus to partition and slice the new disk. If you can not use the menus, use **fdisk**(1M) to partition the disk, if necessary; use **prtvtoc**(1M) and **edvtoc**(1M) to slice it.

Step 6: Use the **sysadm** menus to make file systems on the new disk. If you can not use the menus, use **mkfs**(1M).

Step 7: Restore the files or rebuild the mirror (if you are using Volume Manager).

Chapter 2 Configuring Adapters

This chapter contains the following procedures for adding adapters to your system:

NOTE: For instructions on installing hardware specific to the NCR 345x or 35xx, see *NCR 345x/35xx Unit Installation, Care and Cleaning*.

Configuring Adapters at System Startup

Beginning with Release 2.02 of NCR UNIX SVR4 MP–RAS, the **madf** package allows you display and modify the configurable items for Micro Channel adapters used in the NCR 345x and 35xx. Without the **madf** package, you must shut down the system and boot from the DOS reference diskette to view or change Micro Channel configuration.

If you have made any changes to your Micro Channel configuration, such as adding or removing adapters, the **adf**(1M) server checks for the following types of configuration changes:

- Missing Adapter Description File (ADF)
- Missing adapter
- New adapter
- Kernel/CMOS conflict

To configure adapters during system startup, perform the following steps:

Step 1: Shutdown the system:

```
# shutdown -y -i0 -g0
```

Step 2: Make the desired Micro Channel configuration changes. Follow the procedures later in this chapter or in the *NCR 345x/35xx Unit Installation, Care, and Cleaning* manual.

Step 3: Power on the unit. During start-up diagnostics, the following message appears:

```
Micro Channel CMOS/POS Setup
.
.
Subtest Name: Micro Channel CMOS/POS Setup
Micro Channel Configuration Change Detected-
Please run the Reference Diskette
```

Step 4: Do NOT run the reference diskette! Boot the operating system instead. When the **adf(1M)** server detects the configuration change, it displays a message similar to the following example:

```
Checking Micro Channel board configuration
Missing board(s):
     Sec (Bus 1), Slot 1 'WD EtherCard PLUS/A
        (WD8003E/A or WD8003ET/A'
New board(s):
     Pri (Bus 0), Slot 6 'Systech 6280 DTC
          Board'

Micro Channel configuration changes have been
detected. You may ignore these changes or take
the system to run level 1 to handle the
changes.

Take system to run level 1: (y/n) ?
```

NOTE: If you want to change the Micro Channel configuration, you must respond within 20 seconds. Otherwise, the system continues to boot; you may change the Micro Channel configuration later, either through the **sysadm** menus or as described in the "Configuring Adapters During System Operation" section of this chapter.

Step 5: Enter y to continue with the configuration. Then follow the procedures for the particular type of configuration change you are making.

Missing ADF

The ADF is an ASCII file that provides descriptions of the bit meaning for the Programmable Options Select (POS) registers supported by an adapter. These registers contains option bits for selecting hardware resource options such as interrupt level, memory address, and I/O address. When you purchase a new adapter, a flex disk containing the ADF for an adapter is usually included with the adapter.

When **adf**(1M) detects a missing ADF, it displays a message similar to the following example:

```
Missing ADF File Handling Options:
  1  ignore  Ignore this change for now
  2  import  Import missing ADF file from a DOS
             Diskette
  3  manual  Take system to run level 1 for manual
             configuration

Select action to take: (default: ignore) [?,??,q]:
```

The system disables any boards for which there is no ADF when configuration changes are made; therefore, you should import missing ADFs whenever possible. Perform the following steps to import missing ADFs:

Step 1: Enter **2** to import the missing ADF. The following message appears:

```
Put the DOS-format floppy diskette containing
the @67el.ADF file in the floppy drive.

Is the diskette ready? (default:y) [y,n,?,q]
```

Step 2: Insert the diskette containing the ADF in the flex drive and press RETURN. When the ADF is copied to the system, the following message appears:

```
ADF file @67el/ADF successfully copied to
/etc/conf/adf.d
```

Drivers that support the Micro Channel configuration utilities automatically install the ADF associated with the driver as part of the the driver package installation. You need only add ADFs for drivers that do not support the Micro Channel configuration utilities.

Missing Adapter

The **adf**(1M) server detects a configuration change whenever one or more adapters that used to be in the system have been removed. The Micro Channel slots that the adapters occupied may now either be empty or occupied by another adapter type. When this configuration change occurs, the Micro Channel configuration utility must clear the saved POS register values form CMOS for the missing adapters.

The message that appears depends on whether the driver that controls the adapter takes advantage of the Micro Channel configuration utilities. Drivers for the following adapters currently support the Micro Channel configuration utilities:

- SCSI
- 8-port and 16-port serial controllers
- Ethernet
- Token–Ring
- DTC

Driver Does Not Support Micro Channel Configuration Utilities

If the driver that controls the adapter does not support the Micro Channel configuration utilities, the following message appears:

```
Missing Board Handling Options:
  1  ignore   Ignore this change for now
  2  remove   Remove this board from the CMOS
              configuration
  3  manual   Take system to run level 1 for manual
              configuration

Select action to take: (default: remove) [?,??,q]:
```

Enter **2** to remove the adapter from the CMOS configuration. Since the driver has not been updated to use the Micro Channel configuration utilities, only the CMOS configuration is affected. The kernel *sdevice* file that might be associated with the adapter can not be determined and is not modified.

Driver Supports Micro Channel Configuration Utilities

If the driver that controls the adapter has been updated to support the Micro Channel configuration utilities, the following message appears:

```
Missing Board Handling Options:
  1  ignore   Ignore this change for now
  2  remove   Remove this board from the CMOS
             configuration, but leave kernel resources
              allocated
  3  delete   Remove this board from the CMOS
             configuration and free kernel resources
              allocated to this board
  4  manual   Take system to run level 1 for manual
              configuration

Select action to take: (default: remove) [?,??,q]:
```

Perform the following steps to remove or delete the adapter from CMOS configuration:

Step 1: Enter **3** if you wish to remove the adapter from the system completely and permanently. If you only wish to remove the adapter for a short time, enter **2** to leave kernel resources allocated and avoid a kernel relink. If you enter **3**, the following message may appear:

```
WARNING: Changes have been made to the kernel
configuration files that require a kernel
relink to take effect.

Relink kernel now? (default:y) [y,n,?,q]
```

Step 2: Enter **y** to relink the kernel to free the *sdevice* resources (interrupt, memory space, etc.) associated with the missing adapter.

New Adapter

The **adf(1M)** server detects a configuration change when one or more new adapters has been added to the system and are not currently configured in CMOS. When **adf(1M)** detects a new adapter, it displays the following message:

```
New Board Handling Options:
  1  ignore  Ignore this change for now
  2  add     Add this board to CMOS and to the kernel
             configuration
  3  manual  Take system to run level 1 for manual
             configuration

Select action to take: (default: add) [?,??,q]:
```

Perform the following steps to configure the new adapter:

Step 1: Enter **2** to configure the new adapter. The Micro Channel configuration utilities configure the system resources to non-conflicting values; adapter-specific configurable items are set to the first choice. You may want to use the **sysadm** menus to ensure that the selections chosen are correct.

Once the new adapter is configured, the following message may appear:

```
WARNING: Changes made to the kernel or CMOS co
nfiguration require a shutdown and reboot of t
he system to take effect.

Shutdown system now? (default:y) [y,n,?,q]
```

Step 2: Enter **y** to shutdown the system. When you add or change board settings, the utilities only make the change in CMOS and do not affect the currently running adapters. When you reboot the system, the firmware loads the new settings.

Kernel/CMOS Conflicts

The **adf**(1M) server detects a configuration change when the kernel *sdevice* file configuration does not match the CMOS configuration for one or more adapters. This conflict usually occurs if adapters are configured with the reference diskette instead of the Micro Channel configuration utilities.

There are two types of mismatches. The first occurs if there is an *sdevice* entry for the adapter, but the settings of the system resources do not match the setting for these resources in CMOS. When **adf**(1M) detects this type of mismatch, it displays the following message:

```
Configuration Mismatch Handling Options:
  1  ignore   Ignore this change for now
  2  change   Change kernel configuration to match
              CMOS
  3  manual   Take system to run level 1 for manual
              configuration

Select action to take: (default: change) [?,??,q]:
```

In this case, enter **2** to change the kernel configuration to match the CMOS settings.

The other type of mismatch occurs if an adapter is configured in CMOS, but no *sdevice* entry exists for the board. When **adf(1M)** detects this type of mismatch, it displays the following message:

```
Configuration Mismatch Handling Options:
  1  ignore   Ignore this change for now
  2  add      Add this board to the kernel
              configuration
  3  manual   Take system to run level 1 for manual
              configuration

Select action to take: (default: add) [?,??,q]:
```

In this case, enter **2** to add the adapter to the kernel configuration.

For both types of mismatch, the following message then appears:

```
WARNING: Changes have been made to the kernel
configuration files that require a kernel relink to
take effect.

Relink kernel now? (default:y) [y,n,?,q]
```

Enter **y** to relink the kernel so that the changes will take effect.

Flex Disk Recovery

If you make a configuration change that affects the SCSI adapter which controls the boot device, it may be impossible to boot the system from hard disk. To recover from this condition, perform the following steps:

Step 1: Insert the first boot flex diskette (volume 1) into the flex disk drive.

Step 2: Reboot the system from the flex diskette.

Step 3: When instructed, insert the second boot flex diskette (volume 2) into the flex disk drive and press RETURN.

Step 4: When the following message appears, enter **4** to configure adapters:

```
Select one of the following:
    1. Continue Installation
    2. Perform System Maintenance
    3. Perform System Restore
    4. Perform Micro Channel Configuration
Type selection number, then press ENTER.>
```

Step 5: When instructed, insert the Micro Channel Configuration diskette into the flex disk drive. Normally, the root device is not available. If it is, however, you are prompted to mount the root file system. Answer **y** to use the *sdevice* files on the root disk.

At this point, the **sysadm** menus for configuring adapters appear. See the *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface* manual for instructions on using these menus.

Minimize the changes you make in the recovery environment. These utilities can not relink the kernel to match configuration changes and can not handle some of the resource conflicts automatically. Only make the changes necessary to boot the system.

Configuring Adapters During System Operation

If you have made any Micro Channel adapter configuration changes to your system, the **adf**(1M) server detects these changes during system startup and asks whether you want to reconfigure the system. If you do not respond in the allocated time (20 seconds), or if you choose the **ignore** selection, the system boots without making the necessary configuration changes.

Once the system boots, you can use either the **sysadm** menus or the **idadf**(1M) utility to display or change the Micro Channel adapter configuration. You can use **idadf**(1M) to perform any of the following tasks:

- View the current Micro Channel adapter configuration
- View a summary of system resource setting for all configured adapters
- View the configurable settings for an adapter
- View the choices for a configurable item on an adapter
- Display help for a configurable item on an adapter
- Change a configurable item for an adapter
- Write changes to CMOS
- Configure an adapter automatically

Viewing Current Configuration

To display the current Micro Channel adapter configuration for your system, enter the following command:

```
# /etc/conf/bin/idadf -S
```

A display similar to the following example appears:

```
0) Pri 0000 Internal Options
1) Pri ffff Empty
2) Pri 0092 NCR 53C700 SCSI MCA Adapter
3) Pri 0092 NCR 53C700 SCSI MCA Adapter
4) Pri 67e1 NCR 8 Port Serial Controller/MC
5) Pri ffff Empty
6) Pri 0280 Systech 6280 DTC Board
7) Pri ffff Empty
8) Pri 6fc0 WD EtherCard PLUS/A
1) Sec ffff Empty
2) Sec ffff Empty
3) Sec 0092 NCR 53C700 SCSI MCA Adapter
4) Sec ffff Empty
5) Sec 621c NCR 16 Port Serial Controller/MC
6) Sec ffff Empty
7) Sec 01ba NCR 53C710 SCSI MCA Adapter
8) Sec 6fc0 WD EtherCard PLUS/A
```

This display shows the slot number, the bus name (if necessary), the four-digit hexadecimal adapter ID, and the adapter name. The bus names are **Pri** for the Primary Micro Channel (bus 0) and **Sec** for the Secondary Micro Channel (bus 1).

Viewing Current Resource Summary

To display a summary of the current system resource settings for all configured adapters, enter the following command:

```
# /etc/conf/bin/idadf -L
```

A display similar to the following example appears:

```
2) Pri NCR 53C700 SCSI MCA Adapter   14 1200-12ff       c2000-c3fff       1
       Kernel Board State: Ok

3) Pri NCR 53C700 SCSI MCA Adapter   14 2000-20ff       c4000-c5fff       3
       Kernel Board State: Ok

4) Pri NCR 8 Port Serial Controller/ 11    -            d4000-d5fff       -
       Kernel Board State: Ok

6) Pri Systech 6280 DTC Board        10    -         8cd00000-8dc1ffff    -
       Kernel Board State: Ok

8) Pri WD EtherCard PLUS/A            3 280-29f         c8000-cbfff       -
       Kernel Board State: Unknown

3) Sec NCR 53C700 SCSI MCA Adapter   14 400-4ff         c6000-c7fff       -
       Kernel Board State: Ok

5) Sec NCR 16 Port Serial Controller 11    -            d8000-dbffff      -
       Kernel Board State: Ok

7) Sec NCR 53C710 SCSI MCA Adapter   14 800-8ff         ce000-cffff       4
       Kernel Board State: Ok

8) Sec WD EtherCard PLUS/A            3 280-29f         d0000-d3fff       -
       Kernel Board State: Unknown
```

This display shows the slot number, bus name (if necessary), the adapter name, the interrupt level, the I/O address range, the memory address range, and the arbitration level assigned to each configured adapter in the system. If the adapter does not use a particular resource type, the field for that resource contains a dash (–). The bus names are **Pri** for the Primary Micro Channel (bus 0) and **Sec** for the Secondary Micro Channel (bus 1).

Kernel board states are as follows:

Ok
: The CMOS and kernel configuration for this adapter match.

Unconfigured
This adapter is configured in CMOS, but there isno kernel *sdevice* file entry for the adapter.

Mismatch
This adapter is configured in CMOS and there is a kernel *sdevice* file entry for the adapter, but some of the resource settings do not match the CMOS values.

Unknown
The kernel driver for this adapter did not supply the necessary configuration files to determine which *sdevice* file corresponds to this adapter.

Viewing Configurable Items for an Adapter

To display the configurable items for a particular adapter, enter the following command:

```
# /etc/conf/bin/idadf -I -sx -By
```

The –s –**B** options specify the adapter in slot *x* of bus *y*. If you do not specify the –**B** option, the default is bus 0. A display similar to the following example appears:

```
# /etc/conf/bin/idadf -I -s6 -B0
0) Micro Channel Memory Address - 8DC00000h
1) Interrupt Level - Level 10
2) Micro Channel Data Bus Parity - Enabled
3) Network Node ID - F0h
4) Host Handshake - Enabled
5) Ram Memory Test - Long Version
6) Real-Time Clock Period - 52 ms
```

This command numbers and displays all the configurable items for an adapter and their current settings. You can use the item number with other commands to manipulate a specific configurable option. This example shows a command that displays the configurable items for the DTC adapter in slot 6 of the Primary Micro Channel (bus 0).

Viewing Choices for a Configurable Item

To display the choices for a configurable item on an adapter, enter the following command:

```
# /etc/conf/bin/idadf -C -ix -sy -Bz
```

The **–i** option specifies the item number (from the **idadf –I** command) for which you want to view the choices. The **–s** and **–B** options specify the slot number and bus number, respectively. If you do not specify the **–B** option, the default is bus 0. A display similar to the following example appears:

```
# /etc/conf/bin/idadf -C -i1 -s6
0)  Level 10
1) *Level 11
2) *Level 12
3) *Level 15
```

This command numbers and displays the options for the specified configurable item. This example shows a command that displays the choices for item 1 (interrupt level) for the DTC adapter in slot 6 of the Primary Micro Channel (bus 0). The asterisks denote options that are in use by other adapters and, therefore, unavailable.

Viewing Help for a Configurable Item

To display help text for a configurable item on an adapter, enter the following command:

```
# /etc/conf/bin/idadf -h -ix -sy -Bz
```

The **–i** option specifies the item number (from the **idadf –I** command) for which you want to view help text. The **–s** and **–B** options specify the slot number and bus number, respectively. If you do not specify the **–B** option, the default is bus 0. A display similar to the following example appears:

```
# /etc/conf/bin/idadf -h -i1 -s6
This field sets the interrupt request level this board
will use. More than one DTC 6280 can share the same
interrupt request level, but normally no other boards
should have the same interrupt request level as the
6280.
```

This example displays the help text for item 1 (interrupt level) for the DTC adapter in slot 6 of the Primary Micro Channel (bus 0).

Changing a Configurable Item

To change the value for a configurable item on an adapter, enter the following command:

```
# /etc/conf/bin/idadf -P -cw -ix -sy -Bz
```

The –c option specifies which value from **idadf –C** you want to select. The –i option specifies the item number (from the **idadf –I** command) for which you want to change the value. The –s and –B options specify the slot number and bus number, respectively. If you do not specify the –B option, the default is bus 0.

NOTE: This command does NOT write changes to CMOS. When you have made all desired changes with this command, use **idadf** –w to write changes to CMOS. See "Writing Changes to CMOS" for more information.

The following example selects choice 4 for configurable item 0 for the adapter in slot 6 of the Primary Micro Channel (bus 0):

```
# /etc/conf/bin/idadf -P -c4 -i0 -s6
```

Writing Changes to CMOS

Once you have made all desired configuration changes, enter the following command to CMOS:

```
# /etc/conf/bin/idadf -w
```

This command returns an exit status of either 0 or 16 for successful completion. An exit status of 16 means that kernel *sdevice* file changes have been made and a kernel relink is required. To determine the exit status, enter the following command:

```
# echo $?
```

Configuring an Adapter Automatically

Instead of individually setting each configurable item for an adapter, you can enter the following command to automatically configure all items for the adapter:

```
# /etc/conf/bin/idadf -a p [-sx [-By]]
```

The **–a p** option specifies a partial configuration. The partial configuration configures only the specified adapters. If you do not specify any adapters, it configures all new, unconfigured adapters.

During automatic configuration, **idadf**(1M) sets system resource configurable items to values that do not conflict with other adapters in the system. It sets other configurable items to the first choice. All changes made with this command are written to CMOS.

The following example automatically configures the adapter in slot 4 on the Secondary Micro Channel (bus 1):

```
# /etc/conf/bin/idadf -a p -s4 -B1
```

This command returns an exit status of either 0 or 16 for successful completion. An exit status of 16 means that kernel *sdevice* file changes have been made and a kernel relink is required. To determine the exit status, enter the following command:

```
# echo $?
```

Installing a Serial Controller

Each serial controller provides additional serial connections to your system. To install the hardware and configure the system to use this controller, use the following procedure.

Before You Start

- Make sure you have a backup disk to serve as the working copy of the reference disk. This disk must be writable (NOT write protected). Also, make sure you label the disk as the working copy of the reference disk.
- Determine where the Serial Controller board is going to be installed; it can be installed in any empty expansion slot.

Preparing the System

Step 1: Shut down the system by entering the following command:

```
# shutdown -i0 -g0 -y
```

Step 2: Turn the power switch on the console and on the system to the OFF position.

Step 3: Disconnect the power cord for the system from the rear panel of the system or from the wall outlet.

Step 4: Install the Serial Controller in your system as instructed in *NCR 345x/35xx Unit Installation, Care and Cleaning.*

Step 5: Reboot your system from the reference diskette and make a working copy of the diskette.

Reconfiguring the System

Installing Configuration Files

Step 1: Press the ESC (escape) key as many times as necessary to return to the main menu.

Step 2: Select **Configuration.** Then, select **Install** or **FileCopy**.

Step 3: When instructed to insert the new adapter diskette, remove the working copy of the reference disk, insert the ACL MC Option Diskette, and press RETURN.

The files are automatically copied from the Option Diskette into system memory.

Step 4: Remove the ACL MC Option Diskette, insert the working copy of the reference disk, and press RETURN.

The files are copied to the reference disk.

Step 5: When the **Install Complete** message appears, press RETURN to return to the Configuration menu.

Changing the Configuration

Step 1: Select **Change** from the choices on the Configuration menu.

Step 2: Make sure the board is added correctly (has the correct slot number).

For example, the following information is displayed if the board is in slot 3:

```
SLOT 3: NCR SERIAL CONTROLLER/MC
```

Step 3: Highlight the slot number of the Serial Controller board to be changed and press the RETURN key to display the current values.

```
Dual Port Address    0d0000H
Enable Interrupts    Disabled
Interrupt Level      IRQ 3
```

Warning

Make sure the base addresses are unique. Also, make sure the interrupts are enabled.

Step 4: Highlight the interrupt level and press the RETURN key to display a list of valid values.

Only four (4) interrupt levels (IRQ values) are available: 3, 9, 10, 11. All Serial Controller boards must have the same IRQ value. You should specify an IRQ value of **11** because the default IRQ value used when you install the **tty** software package is **11**. If you use another IRQ value, make sure the **tty** package is installed with the new value.

Step 5: Highlight the new interrupt level, such as **IRQ 11,** and press the RETURN key to select that value.

Step 6: Press function key 10 (F10) to save the new interrupt level.

Step 7: If necessary, use the arrow keys and the RETURN key to enable the board and to change any other information about the board, such as the base address.

The recommended base address for each board appears in Figure 2–1.

	Dual Port (base) Address	Enable Interrupts	Interrupt Level
first board	0d0000H	Enabled	IRQ 11
second board	0d2000H	Enabled	IRQ 11
third board	0d4000H	Enabled	IRQ 11
fourth board	0d6000H	Enabled	IRQ 11
fifth board	0d8000H	Enabled	IRQ 11
sixth board	0dA000H	Enabled	IRQ 11

Figure 2–1: Board Base Addresses and Interrupt Levels

The values shown in Figure 2–1 for the interrupt levels are the same as the default values used when you install the **tty** package (driver and software).

There are no default values for board addresses; they are chosen based on the values other devices are already using. The first board installed during package installation contains the first block of ports/devices. The next board contains the next block, and so on.

Backing Up the Configuration

Step 1: After you configure the board(s), return to the **Configuration** menu and select **Backup.**

A configuration backup file is written to the Reference Disk so it can be used to restore the configuration information, if necessary.

Step 2: When the **Backup Complete** message appears, remove the working copy of the reference disk.

Step 3: Press the ESC (escape) key as many times as necessary to exit the system Setup; the system reboots and loads the operating system.

NOTE: To use terminals on the Serial Controller, follow the procedure in the chapter "Configuring Serial Devices."

Installing an Ethernet LAN Adapter

Use the following procedure to install an Ethernet LAN adapter and configure your system to use it.

Before You Start

- Determine where the Ethernet LAN adapter is going to be installed; it can be installed in any empty expansion slot.
- Make sure you have a backup disk to serve as the working copy of the reference disk. This disk must be writable (not write protected). Also, make sure you label the disk as the working copy of the reference disk.

Preparing the System

Step 1: Shutdown the system by entering the following command:

```
#   shutdown  -i0  -g0  -y
```

Step 2: Turn the power switch on the console and on the system to the OFF position.

Step 3: Disconnect the power cord for the system from the rear panel of the system or from the wall outlet. On an NCR 345x or 35xx, flip the circuit breakers and remove the CMOS battery fuse.

Step 4: Install the Ethernet LAN Module as instructed in *NCR 345x/35xx Unit Installation, Care and Cleaning.*

Step 5: Reboot the system from the reference diskette; make a copy of the reference diskette.

Reconfiguring the System

Install Configuration Files

Step 1: Press the ESC (escape) key to return to the main menu.

Step 2: Select **Configuration** and then select **Install** or **FileCopy**.

Step 3: When instructed to insert the new adapter diskette, remove the working copy of the reference disk and insert the superdisk.

The files are automatically copied from the superdisk into system memory.

Step 4: Remove the superdisk and insert the working copy of the reference disk. The files are copied to the reference disk.

Step 5: When the **Install Complete** message appears, press the RETURN key to return to the **Configuration** menu.

Change Configuration

Step 1: Select **Change** from the choices on the Configuration Menu.

Step 2: Make sure the Ethernet LAN adapter has the correct slot number.

For example, the following information is displayed if the Ethernet LAN adapter is in slot 3:

```
SLOT 3: WD EtherCard PLUS/A
(WD8003E/A or WD8003ET/a)
```

Step 3: Use the down arrow key to highlight the slot number for the Ethernet LAN Module and press the RETURN key.

The following information should be displayed:

```
Fixed resources "Fixed resources"
Adapter IO Space IO Base 280h
Shared Ram Space (16K Bytes) Ram Base D0000h
BIOS ROM Space BIOS ROM Disabled Interrupt
Level Level 3
```

Step 4: Highlight the interrupt level, such as IRQ 10, and press RETURN to select that value.

You should select and IRQ value of 10 because 10 is the default IRQ value used with you install the **ild** package. If you use another IRQ value, make sure you install the **ild** package with the same value.

Step 5: If the interrupt level must be changed, highlight the **Interrupt Level** field and press the RETURN key to display a list of valid values.

Step 6: Highlight the new interrupt level and press the RETURN key.

Step 7: Press Function Key 10 (F10) to save the new interrupt level.

Step 8: Press the ESC key to return to the **Configuration** menu.

Back Up the Configuration

Step 1: Select **Backup**.

A configuration backup file is written to the reference disk so it can be used to restore the configuration information, if necessary.

Step 2: When the **Backup Complete** message appears, remove the working copy of the reference disk.

Step 3: Press the ESC key as many times as necessary to exit the system setup; the system is automatically rebooted and the operating system is reloaded.

Preparing Your Software to Use the Ethernet LAN Adapter

Step 1: If you are installing the first Ethernet LAN adapter, enter the **pkginfo** command to make sure the appropriate software package, such as **ild** (Integrated LAN Driver), is installed.

```
# pkginfo
```

If you are installing the second Ethernet LAN adapter, enter the following command to save the current software configuration file.

```
# mv /etc/ild/ildcf /etc/ild/ildcf.old
```

Refer to the *Integrated LAN Driver* book for the procedure to configure the driver and the procedure to use the *ildcf.old* file to update the software configuration.

Step 2: If the software is not on your system, use the **pkgadd** command to install it.

```
# pkgadd -d /dev/rmt/device
```

Adding a SCSI Controller to a Uniprocessor System

Use the following procedure to add an additional SCSI controller board to an NCR 3345, 3445, or 3447 unit. These are the only uniprocessor units that can have additional controller boards. (The first SCSI controller is factory installed in these units.) If your system is an NCR 3450 or 3550, use the procedure in the next section, "Adding a SCSI Controller to a Multiprocessor System."

Before You Start

- Make sure you have a backup disk to serve as the working copy of the reference disk. This disk must be writable (not write protected). Also, make sure you label the disk as the working copy of the reference disk.
- Determine where the SCSI controller is going to be installed. It is recommended that you insert the second SCSI controller in slot 4 so it is next to the first SCSI controller. The first SCSI controller controls the *root* (*/*) device, and it MUST be in slot 5.
- Do NOT insert the SCSI controller until after you modify the operating system SCSI file.

Modifying the SCSI File

Step 1: Put the system in single user mode:

```
# init 1
```

Step 2: Enter the following command:

```
# cd /etc/conf/sdevice.d
```

Step 3: Use an editor, such as **vi**, to view the contents of the *SCSI* file to make sure there is an entry with an interrupt level of **14**. The sixth field is the interrupt level. For example, the *SCSI* file may contain the following lines:

```
SCSI   Y   1   5   1   5    240   25f   0   0
SCSI   N   2   5   1   9    340   35f   0   0
SCSI   Y   1   5   1   14   a00   bff   0   0
```

There is no need to need to change the contents of this file if you want your second SCSI controller to use **14** as its interrupt level. If you want to assign an interrupt level of **9** to the second SCSI controller, change the **N** in the second line to a **Y** and make sure you do not use an interrupt level of **9** for any Serial Controller boards that are going to be configured in the system.

When the system is rebooted, **mkdtab**(1M) or **mktable**(1M) automatically makes the correct device nodes for devices it finds on the new controller. Refer to Appendix B for more information about SCSI device names and minor numbers.

Step 4: Build a new kernel by entering the following command:

```
# /etc/conf/bin/idbuild
```

Preparing the System

Step 1: Shutdown the system by entering the following command:

```
# shutdown -i0 -g0 -y
```

Step 2: Turn the power switch on the console and on the system to the OFF position.

Step 3: Disconnect the power cord for the system from the rear panel of the system or from the wall outlet.

Step 4: Install the SCSI Controller as instructed in *NCR 345x/35xx Unit Installation, Care and Cleaning*.

Step 5: Connect all subsystems to the second SCSI controller and power them on. Reconnect the power cord.

Step 6: Reboot the system from the reference diskette; make a copy of the reference diskette.

Reconfiguring the System

Enable the Second SCSI Controller

Step 1: Select **Configuration**. Do NOT use automatic reconfiguration.

Step 2: Select **Change.**

Step 3: Select **Internal Options** and press the RETURN key.

Step 4: Select **SCSI Boot Slot** and press the RETURN key.

Step 5: Highlight the **Boot off highest SCSI slot** item and press the RETURN key.

Step 6: Press function key 10 (F10) to save the configuration information.

Step 7: Press the ESC key as many times as necessary to return to the **Configuration** menu.

Configure the First SCSI Controller

Step 1: Highlight the slot number of the first SCSI controller:

```
Parallel Bus SLOT 5: NCR SCSI Host Adapter Board.
```

Press the RETURN key to display the default values.

Step 2: Highlight the **Interrupt Level** field and press the RETURN key.

Step 3: Make sure the interrupt level is 5. Then, press the RETURN key.

Step 4: Highlight the **I/O Address Select** field and press the RETURN key.

Step 5: Set the I/O address for the first SCSI controller to 200. Then, press the RETURN key.

Step 6: Make a note of the **Static Ram Starting Address** value for the first SCSI controller.

> **Caution**
>
> Do NOT change the value in the **Host Adapter Board Logical ID** field.

Step 7: Press Function Key 10 (F10) to save the configuration information. Then, press the ESC key.

Configure the Second SCSI Controller

Step 1: Highlight the slot number of the second SCSI controller. For example,

```
Parallel Bus SLOT 4: NCR SCSI Host Adapter Board.
```

Press the RETURN key to display the default values.

Step 2: Highlight the **Interrupt Level** field and press the RETURN key.

Step 3: Highlight one of the interrupt levels to be assigned to the second SCSI controller; make sure the interrupt level for the SCSI controller does not conflict with the interrupt level of any other board in the system. It is recommended that you assign the second SCSI controller an interrupt level of **14**. If you assign an interrupt level of **9** to the second SCSI controller, make sure you do not use an interrupt level of **9** for any Serial Controller boards that are going to be configured in the system. Then, press the RETURN key.

Also make sure the selected interrupt level matches the interrupt level in the *sdevice* file for SCSI.

Step 4: Highlight the **I/O Address Select** field and press the RETURN key.

Step 5: Set the I/O address for the second SCSI controller to 300. Then, press the RETURN key.

The first SCSI controller, which controls the *root* device, must have the lowest I/O address. The second SCSI controller must have a higher I/O address than the first SCSI controller.

Step 6: Make sure the **Static Ram Starting Address** value for the second SCSI controller does not conflict with the value for the first SCSI controller; the addresses must be unique. Make a note of this address.

> **Caution**
>
> Do NOT change the value in the **Host Adapter Board Logical ID** field.

Step 7: Review and save all other system configuration entries to ensure the integrity of the Reference Diskette.

Step 8: Press Function Key 5 (F5) to save and print the new configuration.

Step 9: Press Function Key 10 (F10) to save the configuration information. Then, press the ESC key.

Step 10: A message appears indicating that the SCSI controller is not enabled and asks if you want to enable it. Enter the letter y and then press Function Key 10 (F10).

Save Configuration

Step 1: Select **Backup** to permanently save the configuration information in a file on the reference disk.

The backup file can be used to restore the configuration information if you have a CMOS RAM or other hardware problem.

Step 2: Remove the working copy of the reference disk and press the ESC key as many times as necessary to reboot the system and load the operating system.

Accessing the Second SCSI Controller

Step 1: Use the menu interface to add, slice, etc., each SCSI hard disk connected to the second SCSI controller.

Step 2: Make sure the new devices can be accessed.

NOTE: If a timeout occurs while attempting to access the new devices on the second controller and the message "SCSI Bus Reset Failed" appears on the screen, the interrupt level in the *sdevice* file for *SCSI* and the reference diskette probably do not match. If this error occurs, shut the system down, remove the second SCSI controller, and repeat the entire procedure, making sure the interrupt level in the reference diskette matches the interrupt level in the *sdevice* file for *SCSI*.

Adding a SCSI Controller to a Multiprocessor System

Use the following procedure to add an additional SCSI controller board to an NCR 3450 or 3550 unit. These are the multiprocessor units that can have additional controller boards. (The first SCSI controller is factory installed in these units.) If your system is an NCR 3345, 3445, or 3447, use the procedure in the previous section, "Adding a SCSI Controller to a Uniprocessor System."

Before You Start

- Make sure you have a backup disk to serve as the working copy of the reference disk. This disk must be writable (not write protected). Also, make sure you label the disk as the working copy of the reference disk.
- Do NOT insert the SCSI controller until after you modify the operating system SCSI file.

Modifying the SCSI File

Step 1: Put the system in single user mode:

```
# init 1
```

Step 2: Enter the following command:

```
# cd /etc/conf/sdevice.d
```

Step 3: Use an editor, such as **vi**, to view the contents of the *SCSI* file to make sure there is an entry with an interrupt level of **14**. The sixth field is the interrupt level. For example, the *SCSI* file may contain the following lines:

```
SCSI   Y   1   5   1   5    240   25f   0   0
SCSI   N   2   5   1   9    340   35f   0   0
SCSI   Y   1   5   1   14   a00   bff   0   0
```

There is no need to need to change the contents of this file if you want your second SCSI controller to use **14** as its interrupt level. On the 3450 and 3550 systems, all SCSI controllers can share **14** as the interrupt level; this is the recommended interrupt level.

If you want to assign an interrupt level of **9** to the second SCSI controller, change the **N** in the second line to a **Y** and make sure you do not use an interrupt level of **9** for any Serial Controller boards that are going to be configured in the system.

When the system is rebooted, **mkdtab**(1M) or **mktable**(1M) automatically makes the correct device nodes for devices it finds on the new controller. Refer to Appendix B for more information about SCSI device names and minor numbers.

Step 4: Build a new kernel by entering the following command:

```
# /etc/conf/bin/idbuild
```

Preparing the System

Step 1: Shutdown the system by entering the following command:

```
# shutdown -i0 -g0 -y
```

Step 2: Turn the power switch on the console and on the system to the OFF position.

Step 3: On the NCR 35xx, trip the circuit breakers. On the NCR 345x, turn the battery switch OFF.

Step 4: Install the SCSI Controller as instructed in *NCR 345x/35xx Unit Installation, Care and Cleaning.*

Step 5: Connect all subsystems to the second SCSI controller and power them on. Reconnect the power cord.

Step 6: Reboot the system from the reference diskette; make a copy of the reference diskette.

Reconfiguring the System

Configure the Second SCSI Controller

Step 1: Highlight the **Interrupt Level** field and press the RETURN key.

Step 2: Highlight one of the interrupt levels to be assigned to the second SCSI controller; make sure the interrupt level for the SCSI controller does not conflict with the interrupt level of any other board in the system. The recommended interrupt level is **14**. Then, press the RETURN key. Make sure the selected interrupt level matches the interrupt level in the *sdevice* file for SCSI.

Step 3: Highlight the **I/O Address Select** field and press the RETURN key.

Step 4: Set the I/O address for the second SCSI controller to the next highest free address. Then, press the RETURN key.

By convention, the first SCSI controller, which controls the *root* device, must have the lowest I/O address. Additional SCSI controllers must have higher I/O addresses than the first SCSI controller. The SCSI driver sorts controllers based on I/O address, so it is important not to add controller with an I/O address in the middle of the other SCSI controllers.

Step 5: Make sure the **Static Ram Starting Address** value for the second SCSI controller does not conflict with the value for the first SCSI controller; the addresses must be unique. Make a note of this address.

Step 6: Make sure the **Arbitration** value for the second SCSI controller does not conflict with the value for the first SCSI controller.

Caution

Do NOT change the value in the **Host Adapter Board Logical ID** field.

Step 7: Review and save all other system configuration entries to ensure the integrity of the Reference Diskette.

Step 8: Press Function Key 5 (F5) to save and print the new configuration.

Step 9: Press Function Key 10 (F10) to save the configuration information. Then, press the ESC key.

Step 10: A message appears indicating that the SCSI controller is not enabled and asks if you want to enable it. Enter the letter y and then press Function Key 10 (F10).

Save Configuration

Step 1: Select **Backup** to permanently save the configuration information in a file on the Reference Disk.

The backup file can be used to restore the configuration information if you have a CMOS RAM or other hardware problem.

Step 2: Remove the working copy of the Reference Disk and press the ESC key as many times as necessary to reboot the system and load the operating system.

Accessing the Second SCSI Controller

Step 1: Use the **sysadm** or **osa** menu interface to add, slice, etc., each SCSI hard disk connected to the second SCSI controller.

Step 2: Make sure the new devices can be accessed.

NOTE: If a timeout occurs while attempting to access the new devices on the second controller and the message "SCSI Bus Reset Failed" appears on the screen, the interrupt level in the *sdevice* file for *SCSI* and the reference diskette probably do not match. If this error occurs, shut the system down, remove the second SCSI controller, and repeat the entire procedure, making sure the interrupt level in the reference diskette matches the interrupt level in the *sdevice* file for *SCSI*.

Chapter 3 Configuring Serial Devices

This chapter contains the the following information and procedures:

The Service Access Facility

The Service Access Facility (SAF) provides an umbrella over the system's external access points and a consistent access mechanism. It consists of two components: the Service Access Controller and the port monitors.

The Service Access Controller

The Service Access Controller (SAC) is the administrative point of control for all port monitors and, therefore, for all ports on the system. The SAC manages port monitors by means of a **sac** process, the */etc/saf/_sysconfig* system configuration script, the */etc/saf/_sactab* administrative file, and the **sacadm**(1M) command.

Port Monitors

Port monitors operate under control of the SAC and may be port monitors for TTY ports (**ttymon** processes) or network listener daemons (**listen**).

The TTY ports are managed by a **ttymon** process, the administrative file */etc/saf/pmtag/_pmtab*, the **sttymon**(1M) command, and the */etc/ttydefs* file.

This chapter describes only the **ttymon** port monitors. For more information about the network port monitors, refer to **listen**(1M) in the *Reference Manual.*

The Service Access Controller

What the SAC Does

The Service Access Controller (SAC) maintains the port monitors on the system in the state specified by the system administrator. It accomplishes this task by performing the following functions:

- Customizing its own environment
- Starting the appropriate port monitors
- Polling its port monitors and initiating recovery procedures when necessary

The **init** process is a general process spawner that is invoked as the last step in the boot procedure. It starts the SAC. The SAC then customizes its own environment by invoking the per–system configuration script *(/etc/saf/_sysconfig)*. Next, it reads its administrative file *(/etc/saf/_sactab)* to determine which port monitors are to be started. For each port monitor it starts, the SAC interprets the port monitor's configuration script (*/etc/saf/pmtag/_config*) if it exists. Finally the SAC starts each port monitor using the command specified in the *_sactab* administrative file (for example, **ttymon**). Each port monitored by a port monitor is specified in that port monitor's administration file (*/etc/saf/pmtag/_pmtab*).

Once the port monitors are running, the SAC polls them periodically for status information. The SAC restarts a failed port monitor if a non–zero restart count was specified for the port monitor when it was created.

The SAC Administrative File

The SAC administrative file, */etc/saf/_sactab*, contains information about all the port monitors for which the SAC is responsible. Each entry in the SAC administrative file contains the following information:

PMTAG
: A unique tag that identifies a particular port monitor. The system administrator is responsible for naming a port monitor. This tag is then used by the SAC to identify the port monitor for all administrative purposes. PMTAG may consist of up to 14 alphanumeric characters.

PMTYPE
: The type of the port monitor. In addition to its unique tag, each port monitor has a type. The type identifies a group of port monitors that are different invocations of the same entity. **ttymon** and **listen** are examples of valid port monitor types. The type facilitates administering groups of related port monitors. If no type is designated, the system administrator has no way of knowing which port monitor tags correspond to port monitors of the same type. PMTYPE may consist of up to 14 alphanumeric characters.

FLGS
: The flags that are currently defined are:

- d — When started, do not enable the port monitor
- x — Do not start the port monitor

If no flag is specified, SAC takes the default action. By default, SAC starts and enables a port monitor

RCNT
The number of times a port monitor may fail before SAC places it in a failed state. Once a port monitor enters the failed state, the SAC does NOT try to restart it. If a count is not specified when the entry is created, this field is set to 0. A restart count of 0 indicates that the port monitor is not to be restarted when it fails.

COMMAND
A string representing the command that starts the port monitor. The first component of the string, the command itself, must be a full pathname.

The sacadm Command

The sacadm(1M) command adds, removes, or modifies entries in the SAC administrative file.

The Port Monitor Administrative File

Each port monitor has its own administrative file, */etc/saf/pmtag/_pmtab*, where *pmtag* is the unique tag identifying the particular port monitor. The *pmtag* is also the first field of the corresponding entry in the */etc/saf/_sactab* file.

Each entry in a port monitor's administrative file defines how the port monitor should treat a specific port and what service is to be invoked on that port. Some fields must be present for all types of port monitors. Each entry must include a service tag to identify the service uniquely and an identity to be assigned to the service when it is started (for example, root).

NOTE: The combination of a service tag and a port monitor tag uniquely defines an instance of a service. The same service tag may be used to identify a service under a different port monitor.

Each entry in the port monitor administrative file must contain the following information:

SVCTAG
: A unique tag that identifies a service. This tag is unique only for the port monitor through which the service is available. Other port monitors may offer the same or other services with the same tag. A service requires both a port monitor tag and a service tag to identify it uniquely. SVCTAG may consist of up to 14 alphanumeric characters.

FLGS
: The flags with the following meanings may currently be included in this field:

- x — Do not enable this port. By default, the port is enabled.

- u — Create a utmp entry for this service. By default, no utmp entry is created.

Note that port monitors may ignore the u flag if creating a utmp entry for the service is not appropriate to the manner in which the service is to be invoked. Some services may not start properly unless utmp entires have been created for them (for example, login).

ID
: The identity under which the service is to be started. The identity has the form of a login name as it appears in */etc/passwd.*

PMSPECIFIC
: Examples of port monitor–specific information are addresses, the name of a process to execute, or the name of a STREAMS pipe to pass a connection through.

COMMENT
A comment associated with the service entry.

The pmadm Command

The **pmadm(1M)** command adds, removes, or modifies entries in the */etc/saf/pmtag/_pmtab* file.

NOTE: To maintain the integrity of the system, you should make changes in the SAC and port monitor administrative files with the **sacadm** and **pmadm** commands and not by editing the files. The SAC does not recognize changes in some of the fields in these files unless they are made using the appropriate administrative command. Editing the file directly can lead to unexpected results.

The Port Monitor ttymon

What ttymon Does

The port monitor **ttymon** performs three main functions:

- Initializing and monitoring TTY ports
- Setting terminal nodes and line speeds for each port it monitors
- Invoking the service associated with a given port whenever it receives a connection request on that port

Each instance of **ttymon** has its own */etc/saf/pmtag/_pmtab* administrative file that specifies the ports to monitor and the services associated with each port. The file contains a ttylabel field that refers to a speed and TTY definition in the */etc/ttydefs* file. See **ttyadm**(1M) for a description of the information specific to **ttymon** that is contained in a **ttymon** administrative file.

When a **ttymon** port monitor is started, it initializes all ports specified in its administrative file, pushes the specified STREAMS modules onto the ports, sets speed and initial **termio**(7) settings, and writes the prompt to the port. It then waits for user input.

A connection request is successful when at least one non–break character followed by a newline character is received from the port. If the service to be invoked is **login**, the newline character is preceded by the user's login name. A newline character is not recognized unless the line speed of the port and the line speed of the device connected to the port are the same.

If an unreadable prompt is printed on the terminal, the user sends a BREAK to indicate that the port and device line speeds are not compatible. Each break indication causes **ttymon** to hunt to the next ttylabel in */etc/ttydefs*, adjusting its **termio(7)** values and reissuing the prompt.

On successful completion of the connection request, **ttymon** interprets the per–service configuration script, if one exists. It then invokes the service associated with the port. This service can be any service configured by the system administrator. A typical example is **login**.

Relationship Between ttymon and Port Monitors

There can be one or many **ttymon** port monitors, each identified by a unique *pmtag*. A port monitor can monitor multiple ports.

A port can have one and only one service associated with it. Each port, and its associated service, is identified by a service tag, *svctag*. For each port monitored by the same port monitor, the service tag must be unique.

When the SAC starts a port monitor, the port monitor reads its administrative file, which contains information about what ports to monitor and what service (process) is associated with each port.

Figure 3–1 shows an example of the relationship between ttymon and port monitors.

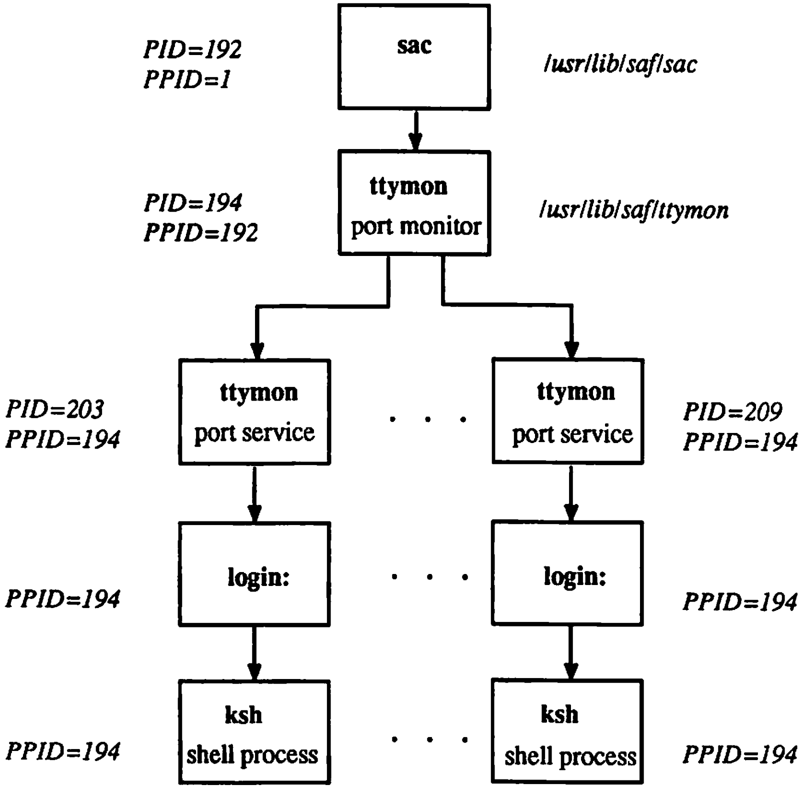


Figure 3–1: Example of **sac, ttymon, login,** and **shell** Relationships

Log Files

You can examine two log files for clues to problems related to **ttymon** port monitors or ports monitored by **ttymon** port monitors:

- */var/saf/_log* — Records abnormal port monitor behavior
- */var/saf/pmtag/log* — Contains records for each **ttymon** port monitor

To list the most recent 25 entries in the *_log* file, enter the following command:

```
# tail -25 /var/saf/_log
```

Periodically, log files should be cleared or truncated. The script */usr/lib/saf/logchecker*, called from the root user's crontab entry, can be configured to maintain SAF and **ttymon** log files

Terminal Line Settings

Each **ttymon** port monitor started by the SAC looks in its administrative file for the TTY ports to initialize. For each TTY port initialized, **ttymon** searches the *ttydefs* file for the information it needs to set terminal modes and line speeds. Then **ttymon** waits for service requests. When it receives a service request, **ttymon** executes the command (usually **login**) associated with the port that received the request. This command is contained in the entry for the port in the port monitor's administrative file.

The ttydefs File

The */etc/ttydefs* file is an administrative file used by **ttymon**. The file defines speed and terminal settings for TTY ports as follows:

ttylabel
: When ttymon initializes a port, it searches the *ttydefs* file for the entry that contains the **termio**(7) settings for that port. The correct entry is the one whose *ttylabel* matches the *ttylabel* for the port. The *ttylabel* for the port is part of the *pmspecific* information included in ttymon's administrative file. By convention, *ttylabel* identifies a baud rate (for example, 1200), but it does not have to.

initial-flags
: Contains the **termio**(7) options to which the terminal is initially set. *initial-flags* must be specified using the syntax recognized by the **stty**(1) command.

final-flags

Contains the **termio**(7) options set by **ttymon** after a connection request has been made and immediately before invoking a port's service. Final flags must be specified using the syntax recognized by **stty**.

autobaud

Autobaud is a line-speed option. When autobaud is used instead of a baud rate setting, **ttymon** determines the line speed of the TTY port by analyzing the first carriage return entered and sets the speed accordingly. If the *autobaud* field contains the character A, the autobaud facility is enabled. Otherwise autobaud is disabled.

nextlabel

If the user indicates (by sending a BREAK) that the current *ttydefs* entry does not provide a compatible line speed, **ttymon** searches for the *ttydefs* entry whose *ttylabel* matches the *nextlabel* field. **ttymon** then uses that field as its *ttylabel* field. A series of speeds is often linked together in this way into a closed set called a hunt sequence. For example, 4800 may be linked to 1200, which in turn is linked to 2400, which is finally linked to 4800.

All **termio**(7) settings supported by the **stty** command are supported as options in the *ttydefs* file.

NOTE: The format of the */etc/ttydefs* file may change in future releases. For continuity across releases, use the **sttydefs**(1M) command to access this file.

Configuring a Serial Port for a Terminal

Perform this procedure to use a terminal on the serial port located on the back of your machine. The examples shown in the procedure are for a 4970 terminal emulating a DEC vt100.

Procedure

Step 1: You do not need to perform this step if you already have a **sac** process and a master **ttymon** process running on your system.

Enter the **sacadm** command to cause **sac** to start a port monitor (master **ttymon** process). For example, to start a port monitor called *pm1*, enter:

```
# sacadm -a -p pm1 -t ttymon -c \
/usr/lib/saf/ttymon  -v  1
```

Step 2: Enter the **pmadm** command to start a port service (secondary **ttymon** process) for the terminal connected to the serial port.

The following command causes the master port monitor *pm1* to start a port service for the first (or only) serial port, */dev/term/00*:

```
# pmadm -a -p pm1 -s 00 -i  root \
-f u -v  1 -m "`ttyadm -h -d /dev/term/00 \
-l  9600  -s /usr/bin/login`"
```

NOTE: The character after the beginning double quote (") character and before the **ttyadm** command must be a grave (accent) (‘) character, NOT a single quote (’) character. Also, the character after the **login** field and before the ending double quote (") character is a grave (accent) character.

Step 3: Using a system editor (such as **vi**), place a line identifying the terminal and mode in the */etc/ttytype* file. For example:

```
vt100  tty00
```

Configuring a Serial Port for a Modem

Perform this procedure to use a modem on the serial port located on the back of your machine. The examples shown in the procedure are for a 2400 baud Hayes–compatible modem, with the port available for bi-directional use.

Before You Start

Make sure the switches on the front and back of the modem are set correctly. For more information about switch settings, refer to the owner's manual for the modem and to the *NCR 345x/35xx Unit Installation, Care and Cleaning* book.

Both the System 3000 and the modem must be configured with the same characteristics. Generally, these characteristics are:

- Baud Rate – Use the speed of the modem
- Data Bits – 8 bit data characters (if 7 bit, use even parity)
- Parity – No
- Stop Bits – 1
- Duplex – Full

Procedure

Step 1: You do not need to perform this step if you already have a **sac** process and a master **ttymon** process running on your system.

Enter the **sacadm** command to cause **sac** to start a port monitor (master **ttymon** process). For example, to start a port monitor called *pm2*, enter:

```
# sacadm -a -p pm2 -t ttymon -c \
/usr/lib/saf/ttymon  -v  1
```

Step 2: Using a system editor, edit the */etc/uucp/Dialers* file to include entries for the modem. For example:

```
hayes =,-, "" \M\dAT\r\c OK\r \EATDT\T\r\c CONNECT \m\c

clocal "" "" \M\c
```

For more information about the escape sequences, refer to the section "Writing Dialing Information" in the "Managing BNU" chapter.

Step 3: Using a system editor, edit the */etc/uucp/Devices* file to place entries for the modem at the end of the file. For example:

```
ACU term/00,M - 2400 Hayes
Direct term/00,M - 2400 clocal
```

Step 4: Enter the **pmadm** command to start the port service for the modem connected to the serial port.

The following command causes master port monitor *pm2* to start a port service for the modem connected to the first or only serial port, */dev/term/00*.

```
# pmadm -a -p pm2 -s 00 -i  root \
-f  u -v  1 -m  "'ttyadm -bhrl -o6 -d \
/dev/term/00 -l  2400NP  -s /usr/bin/login'"
```

NOTE: The character after the beginning double quote (") character and before the **ttyadm** command must be a grave (accent) (‘) character, NOT a single quote (') character. Also, the character after the **login** field and before the ending double quote (") character is a grave (accent) character.

The tty label for a modem port should refer to a line setting with the hangup on close (**hupcl**) option. You can verify this setting with the following command:

```
# sttydefs -l ttylabel
```

The **hupcl** option should be listed in the final flags for the given label.

Configuring the EIA Mode for a Serial Port

Why/When

If your Serial Adapter is a 16–Port Multiport Adapter with selectable EIA232/422 interface, you can configure each port to use either EIA232 or EIA422 mode. The default mode for all ports is EIA232.

Which mode you use depends on the device using the port. See your device documentation to find out which mode your devices require.

Identifying the Current EIA Mode

To display the current EIA mode for a port, enter the following command:

```
# /sbin/ttyeia -c -d device
```

Device is the pathname of the device connected to the port (for example, */dev/tty/s00*).

Changing the EIA Mode

To change the EIA mode for a port, enter the following command:

```
# /sbin/ttyeia -s mode -d device
```

Mode is the desired EIA mode (232 or 422) and *device* is the pathname of the device connected to the port (for example, */dev/tty/s00*).

Changing the Default EIA Mode

If you want certain ports to always use EIA422 mode without having to change the mode every time the system boots, you can change the mode automatically at boot time. To change modes automatically, edit the */sbin/ttyseteia* shell script, which runs whenever the system boots. This file has the following format:

```
#/sbin/ttyeia -s 422 -d /dev/term/s00
#/sbin/ttyeia -s 422 -d /dev/term/s01
          .
          .
#/sbin/ttyeia -s 422 -d /dev/term/s63
```

To change any ports to EIA422 mode automatically when the system boots, simply remove the comment symbol (#) at the beginning of the line for the desired ports.

Configuring a Second Serial Port

The serial port on the NCR 33xx and 34xx (except models 3320 and 3340) allows you create two serial ports by attaching to the serial port on the back of the unit a special cable that forks to provide two serial port connections. The order number for this cable is documented in the *NCR 345x/35xx Unit Installation, Care and Cleaning* book.

After you attach the special cable to the serial port, you must configure the system software to support a second serial port. Use the following procedure to configure a second serial port.

Before You Start

The second serial port on your system must use Interrupt Level 3 which can be used by the **ild** driver; therefore, to use a second serial port, you must ensure that the ILD driver is using an IRQ other than 3. If you do not know a valid, unused interrupt level for the **ild** driver and Ethernet board, boot off the reference diskette and determine one before beginning this procedure.

Procedure

Step 1: Using a system editor such as **vi**, edit the */etc/conf/sdevice.d/asy* file by adding a line for the second serial port to the end of the file. The additional line should be identical to the original line except:

- The interrupt level in the sixth field should be 3.
- The entries in the seventh and eighth fields should be 2f8 and 2ff.

For example:

```
asy   Y   1   6   1   4   3f8   3ff   0   0
asy   Y   1   6   1   3   2f8   2ff   0   0
```

Step 2: Specify a different interrupt level for the **ild** driver by editing the */etc/conf/sdevice.d/ild* file. Change Interrupt Level 3 to another available value. Make sure the value you specify is not already used.

For example, if you use the value *X*, the file could look like the following:

```
ild  Y  1  3  3  X  0  0  0  0
```

Step 3: Edit the */etc/conf/node.d/asy* file to include the device nodes for the second serial port. Use the same major number as for the original serial port, but increase the minor number by one. For example:

```
asy  term/01    c   1
asy  term/01s   c   1
asy  term/01h   c   129
asy  tty01    c   1
asy  tty01s   c   1
asy  tty01h   c   129
```

Step 4: Rebuild the kernel by entering the **idbuild** command.

Step 5: Perform a system shutdown:

```
# shutdown -y -i6 -g0
```

Step 6: Reboot your system from the reference diskette.

Step 7: Using the reference diskette, change the interrupt for the Ethernet board to be the value you specified for the **ild** driver in Step 2.

Step 8: Reboot the system.

Configuring Serial Controller Ports for Terminals

Each 8–Port or 16–Port Serial Controller provides additional serial connections to your system. After installing the controller, you must configure the system to use this controller and add the terminals before you can use them.

Before You Start

- Add the serial controller using the procedure in the chapter "Configuring Adapters."
- Make sure the **terminf** and **tty** software packages are installed by performing the following command or by examining the contents of the */usr/lib/terminfo* directory.

  ```
  # pkginfo
  ```

- If the packages are not installed or if the compiled *terminfo* files do not exist, install the **terminf** and **tty** packages from flex disk or cartridge tape using the following command:

  ```
  # pkgadd -d device_name
  ```

  See Appendix B, "Device Names and Numbers," for an explanation of device names.

NOTE: When you install the **tty** package, the first board is installed by default with an IRQ value of 9. However, if you add more than one board, make sure the IRQ values you enter during the **tty** package installation are exactly the same as the IRQ values you entered when installing the serial controller (see the chapter "Configuring Adapters").

Procedure

Step 1: Connect each terminal to a port (25–pin connector) on the distribution box and turn the power switch on each terminal to the ON position.

Step 2: Put the system in multiuser mode.

Step 3: Enter the **sacadm** command to cause **sac** to start a port monitor (master **ttymon** process). For example, to start a port monitor called *pm1*, enter:

```
# sacadm -a -p pm1 -t ttymon -c  \
/usr/lib/saf/ttymon -v 1
```

Step 4: Enter the **pmadm** command to start a port service (secondary **ttymon** process) for the port.

For example, enter the following command to cause the port monitor called *pm1* to start a port service for the first terminal, */dev/term/s00*, connected to the first port, labelled *01*, on the first board.

```
# pmadm -a -p pm1 -s s00 -i root -f u   \
-v 1 -m  "`ttyadm -h -d /dev/term/s00 \
-l 9600  -s /usr/bin/login`"
```

NOTE: The character after the beginning double quote (") character and before the **ttyadm** command must be a grave or accent (') character, NOT a single quote (') character. Also, the character after the **login** field and before the ending double quote (") character is a grave (accent) character.

Step 5: Press the ENTER key on the terminal added to make sure a login banner is displayed.

Step 6: Test the terminal by logging in to the system as a valid user and performing one or more shell commands.

Step 7: If the terminal works properly, repeat Steps 4 through 6 for all other terminals that are going to use the same port monitor (**ttymon**) process (for example, */dev/term/s00*, */dev/term/s01*, etc.).

Make sure you change the values in the **–s** option field of the **pmadm** command to **s01, s02**, etc. and change the device names to */dev/term/s01*, */dev/term/s02*, etc.

If the terminal does not work properly, check the terminal cable and the terminal setup (baud rate, etc.)

To Define the Terminal Type

If you want to permanently define the terminal type, use an editor to create an entry in the */etc/ttytype* file for the terminal type. For example,

```
vt100   s00
```

Also, make sure the */etc/ttysrch* file contains the following entries in the order shown. The system runs faster if */dev/term* is checked before */dev*.

```
/dev/term
/dev
```

To Examine the Processes on a Terminal

If you want to examine the processes running on a terminal connected to a serial controller board, enter the following command:

```
# ps  -ftterm/s01
```

Configuring a Serial Controller for Modems

Each 8–Port or 16–Port Serial Controller provides additional serial connections to your system. After installing the controller, you must configure the system to use this controller and add the modems before you can use them.

Examples in this section are for a 2400 baud, Hayes–compatible modem. After configuration, the port is available for bi-directional use.

Before You Start

- Add the serial controller using the procedure in the chapter "Configuring Adapters."
- Make sure the **terminf** and **tty** software packages are installed by performing the following command or by examining the contents of the */usr/lib/terminfo* directory.

  ```
  # pkginfo
  ```

- If the packages are not installed or if the compiled *terminfo* files do not exist, install the **terminf** and **tty** packages from flex disk or cartridge tape using the following command:

  ```
  # pkgadd -d device_name
  ```

  See Appendix B, "Device Names and Numbers," for an explanation of device names.

NOTE: When you install the **tty** package, the first board is installed by default with an IRQ value of 9. However, if you add more than one board, make sure the IRQ values you enter during the **tty** package installation **are exactly the same** as the IRQ values you entered when installing the serial controller (see the chapter "Configuring Adapters").

Procedure

Step 1: Connect each modem to a port (25–pin connector) on the distribution box and turn the power switch on each modem to the ON position.

Step 2: Put the system in multiuser mode.

Step 3: You do not need to perform this step if you already have a **sac** process and a master **ttymon** process running on your system.

Enter the **sacadm** command to cause **sac** to start a port monitor (master **ttymon** process). For example, to start a port monitor called *pm1*, enter:

```
# sacadm -a -p pm1 -t ttymon -c \
/usr/lib/saf/ttymon -v 1
```

Step 4: Using a system editor, edit the */etc/uucp/Dialers* file to include entries for the modem. For example:

```
hayes =,-, "" \M\dAT\r\c OK\r \EATDT\T\r\c CONNECT \m\c
clocal "" "" \M\c
```

For more information about the escape sequences, refer to the section "Writing Dialing Information" in the "Managing BNU" chapter.

Step 5: Using a system editor, edit the */etc/uucp/Devices* file to place entries for the modem at the end of the file. For example:

```
ACU term/s00,M - 2400 hayes
Direct term/s00,M - 2400 clocal
```

Step 6: Enter the **pmadm** command to start a port service (secondary **ttymon** process) for the port.

For example, enter the following command to cause the port monitor called *pm1* to start a port service for a 2400 baud modem connected to the first port, labelled *01*, on the first serial controller (port s00).

```
# pmadm -a -p pm1 -s s00 -i root -f u   \
-v 1 -m "`ttyadm -bhrl -o6 -d /dev/term/s00 \
-l 2400NP -s /usr/bin/login`"
```

NOTE: The character after the beginning double quote (") character and before the **ttyadm** command must be a grave or accent (‘) character, NOT a single quote (’) character. Also, the character after the **login** field and before the ending double quote (") character is a grave (accent) character.

Step 7: Test the modem by establishing contact with a remote system and receiving data from or performing shell commands on the remote system.

Step 8: If the modem works properly, repeat Steps 5 through 7 for all other modems that are going to use the same port monitor (for example, */dev/term/s00*, */dev/term/s01*, etc.).

Make sure you change the values in the –s option field of the **pmadm** command to **s01, s02**, etc. and change the device names to */dev/term/s01*, */dev/term/s02*, etc.

If the information to be placed in the *Dialers* and *Devices* files is the same for additional modems, do not repeat Steps 5 and 6.

If the modem does not work properly, check the entries in the */etc/uucp/Devices* and */etc/uucp/Dialers* files, the physical connections, and the switch settings for characteristics (baud rate, etc.). These characteristics must be configured the same way on the modem and the system.

Setting Modem Options

This section describes how to set up a few important modem options after you have configured the modem ports. These steps will prepare your modem for both dial-in and dial-out operation.

Configure older model modems by setting switches on their front or rear panels. Configure later model modems by issuing commands to them through their serial port using the "Call UNIX" utility **cu**(1).

The following modem "AT" commands work on most later model modems, but your modem's command set may differ. Since modem command sets vary, keep your modem's manual nearby. Press the ENTER key to terminate all these commands.

Step 1: If you are configuring a modem connected to */dev/term/00* with **cu**(1), enter the following command to access the modem:

```
cu -lterm/00
```

Step 2: Turn on your modem's local echo:

```
ATE1
```

Step 3: Force your modem to send responses, since most **uucp**(1) and **cu**(1) dialing sequences expect them:

```
ATQ0
```

Step 4: Configure you modem to send words rather than numeric responses:

```
ATV1
```

Step 5: Configure your modem to disconnect after losing Data Terminal Ready (DTR), avoiding unwanted toll charges:

```
AT&D2
```

The command **AT&D3** also works.

Step 6: Configure your modem's Carrier Detect (CD) signal to follow the remote system. This command helps prevent runaway echo and even enhances security.

```
AT&C1
```

Step 7: If your modem's data rate is 9600 bps or higher, enable hardware flow control, also called RTS/CTS flow control. The "AT" command for RTS/CTS flow control is not well standardized. The following list gives the commands for some modems; check your modem documentation for your modem's command.

Modem	AT Command
DSI 9624	AT*F3
Microcom AX/2400	AT\Q3
Practical Peripherals 9600SA	AT&K3
Telebit Trailblazer Plus	AT&K3

NOTE: Software flow control, also called XON/XOFF flow control, is not recommended, since XON/XOFF sequences can appear in binary data.

Step 8: Configure your modem to automatically answer, if desired:

```
ATS1=1
```

The number after the equal sign is the number of rings to wait for.

Step 9: Save the configured settings

```
AT&W
```

Step 10: Exit **cu**(1) by pressing the ENTER key, then typing the following:

```
.
```

Your modem is now permanently set to answer incoming calls.

NOTE: If your modem tends to have runaway echo (its data indicators continue to flicker between calls), turn off your modem's local echo. Repeat the steps above, adding the following command after Step 8:

```
ATE0
```

Enabling Dial-up Password Protection

To implement an extra layer of security, you may enable dial-up passwords on any system ports accessed through the **login**(1) command. You may require the user to enter an additional dial-up password after entering his or her individual password, depending on the user's shell. Dial-up password protection works on network TTY devices as well as on modem ports.

Dial-up passwords are configured by creating and editing two files: */etc/dialups* and */etc/d_passwd*. Since these files are not created during installation, dial-up password protection is initially disabled. To enable dial-up passwords, create both files using your favorite editor.

Creating /etc/dialups

The */etc/dialups* file is an ASCII file that lists the terminal lines that are considered "dial-up lines." Each terminal line entry in this file is contained on a single line.

The following is a typical dialups file:

```
/dev/term/00
/dev/term/01
/dev/term/s00
/dev/term/s01
/dev/nty/000
/dev/nty/001
```

Creating /etc/d_passwd

The */etc/d_passwd* file is an ASCII file that contains the following information on each line:

```
login shell:encrypted password:optional comment
```

The fields for each shell are separated by colons. Each shell must be on a separate line. The following is a typical example:

```
/bin/sh/gCeu3bdAx2:Require dialup password for /bin/sh
/usr/bin/ksh:fMrc6ymG25xfX
/usr/lib/uucp/uucico:qBms8Yz4fp:Even uucp can have one
/usr/bin/sh:aDfv4ceBy3_,B:Default dialup password
```

When you login on one of the lines from */etc/dialups* and your login shell matches an entry in the */etc/d_passwd* file, you are prompted for a dial-up password before you can access the system. You can not access the system if you do not enter the correct dial-up password.

If the encrypted password field is null, no password is required. The password is only required when your login shell matches a shell specified in the */etc/d_passwd* file and you are logging in on a port listed in */etc/dialups*. If your shell is not listed in */etc/d_passwd* but */usr/bin/sh* is listed, then **login**(1) requires you to enter the password associated with the */usr/bin/sh* encrypted password.

Since passwords are encrypted, */etc/d_passwd* can have general read permissions.

Generating Encrypted Passwords

If you have the Security Administration Utilities package (**crypt**) installed on your system, you can use the **makekey**(1) utility to generate encrypted passwords easily. Otherwise, you may use the following procedure:

Step 1: Create a dummy login:

```
# passmgmt -a dummy
```

Step 2: Assign *dummy* a password. Be sure to remember what it is!

```
# passwd dummy
New password: password
Re-enter new password: password
```

Step 3: Get the encrypted password:

```
# awk -F: '$1 == "dummy" { print $2 }' \
</etc/shadow
```

Step 4: Remove the dummy login:

```
# passmgmt -d dummy
```

Step 5: Use your favorite editor to edit the */etc/d_passwd* file. Insert the encrypted password into the second field of the line with the desired shell.

Disabling Dial–up Password Protection

To disable dial-up password protection, remove either */etc/dialups*, */etc/d_passwd*, or both.

Configuring a Serial Port for a Three–Wire Cable

Why/When

If you wish to use a three–wire printer cable, you must set up a port monitor and service to hold the serial printer port open. This procedure enables three–wire printer cables to work with software flow control.

You can also use a three–wire cable for terminals, but you do not need to perform the configuration procedure for terminals. However, the same restrictions apply to terminals as to printers.

Before You Start

Before you configure a serial port for a three–wire printer cable, the **lp** package must be installed on your system and you must have configured your three–wire serial cable as described in *NCR 345x/35xx Site Preparation.*

Restrictions

Three–wire cables can not provide as much functionality as a complete serial cable. You should be aware of the following restrictions:

- The use of three–wire serial cables is supported in NCR UNIX SVR4 operating system software Release 2.02 or greater.
- The device connected to the three–wire serial cable must support software flow control (XON/XOFF). The device can not depend on any signal transitions except Transmit Data and Receive Data.

- Parity must be set correctly. XON/XOFF characters that have wrong parity are not detected.
- Hardware flow control must be turned off for the port using the three–wire serial cable.
- Modems, **cu/uucp** direct connections, and devices such as multiplexors and switches that require DTR (or DSR) transitions can not be connected to three–wire serial cables.
- Because software can not detect when a device is turned off, any output direct to powered–off printers and terminals will be lost. This loss of data should not be considered to be a failure of the system or software.
- There is an increase of spurious characters when terminals are turned on or off.
- System security may be compromised because the terminal service (**ttymon**) does not know when a terminal gets turned off. Therefore, if a terminal is turned off without being logged off, the session remains active and anyone can turn the terminal back on and have access to the system.

Procedure

Perform the following steps to configure a serial port for a three–wire printer cable:

Step 1: Add a new */etc/ttydefs* entry, if necessary.

The */etc/ttydefs* file contains two entries for a serial port for a three–wire printer cable: *9600PR* and *19200PR*. These entries should meet the needs of most systems, but you may have to make some changes to suit your particular environment. If you create a new entry for your environment, make sure that software flow control is turned on, hardware flow control is turned off, and the **hupcl** option is NOT set.

The following is an example */etc/ttydefs* entry for a 9600 baud printer:

```
9600PR: 9600 opost onlcr tab3 ingpar ixon \
-ixany ixoff istrip echo echoe echok isig cs8\
clocal ignbrk -brkint noflsh cread : 9600 \
opost onlcr sane tab3 ignpar ixon -ixany \
ixoff istrip echo echoe echok isig cs8 clocal\
ignbrk -brkint noflsh cread ::9600PR
```

See **stty**(1) in the *NCR UNIX SVR4 MP–RAS Reference Manual* for information about the options supported in the */etc/ttydefs* file.

Step 2: Set up the port monitor and port service with the following command:

```
# setup_port
```

The **setup_port** utility asks you to enter the pathname for the port (for example, */dev/term/00s*) and the baud rate of the printer (for example, *9600*).

Step 3: Configure the printer. You may use the OSA or sysadm menu interface, or you may follow the instructions in the "Using the LP Print Service" chapter. Regardless of which method you use, you MUST configure the printer as follows:

- The printer must use the same port specified for the port service.
- Software flow control (XON/XOFF) must be enabled.
- The printer must use the same baud rate specified in the */etc/ttydefs* entry.
- The following additional **termio** settings must be set for the printer port: *hupcl*, *clocal*, *–brkint*, *noflsh*, and *ignbrk*.

Chapter 4

Using The LP Print Service

This chapter describes the basic administrative tasks required for setting up and running an LP Print Service. It includes the following information:

What is the LP Print Service?

The LP Print Service is a set of software utilities that allows you to send a file or files to be printed while you continue with other work.

What the LP Print Service Does

The LP Print Service performs the following tasks:

- Receiving files users want printed
- Scheduling the work for one or more printers
- Starting programs that interface with the printer(s)
- Filtering files (if necessary) so they are printed properly
- Keeping track of the status of jobs
- Keeping track of forms and print wheels currently mounted and alerting you to mount needed forms and print wheels
- Issuing error messages when problems arise

How You Configure the LP Print Service

You must set up the LP Print Service to make it operational on your system. Although you can do this using the **lpadmin**(1M) command, using the system administrator menus is recommended. For the menu procedures, see the *NCR UNIX SVR4 Administrator Guide: OA&M Menu Interface.*

Configuring the LP Print Service includes:

- Installing any hardware (terminals or printers) for the LP Print Service
- Establishing an interface program for each printer
- Defining the destinations
- Defining any classes

Files and Directories for the LP Print Service

This section describes the files and directories used by the LP Print Service.

/var/spool/lp/fifos/FIFO
A named pipe that all the commands use to send messages to **lpsched**(1M). Any of the LP Print Service commands may write to *FIFO*, but only **lpsched**(1M) may read it.

/etc/lp/default
Contains the name of the system default destination. If this file does not exist or if it is empty, the LP Print Service has no default destination.

/var/lp/logs

This directory contains three log files: *lpsched, lpNet,* and *requests*. The *lpsched* file contains information about when the lp spooler was started and stopped, when a network process for remote printing has been terminated, and other messages about the status of the print spooler. The *lpNet* file contains information about remote printing, including the start/stop times of the lpNet parent, remote connection information, and network error messages. The *requests* file contains information about all print requests, including the file name, user ID, and destination. When the spoolser is invoked with the debug option, exec and messages files are also created in this directory.

/var/spool/lp/system/pstatus

Contains status information for each printer. Entries are added and removed from this file by the **lpadmin**(1M) command, and they are modified by the **cancel**(1), **enable**(1), **disable**(1), and **lpsched**(1M) commands. When the **lpstat**(1) command is invoked with the **–p** option, printer status information is obtained from this file.

/var/spool/lp/temp/.SEQF

Contains four numbers separated by colons (:). The last number is the sequence number of the last request id that was assigned by the **lp**(1) command. Each **lp** request increments the sequence number by the value in the third number. When the sequence number reaches the second number, it is reset to the first number.

/etc/lp/classes

Contains one file for each identified LP Print Service class. (The name of the file is the same as the name of the class.) The file identifies each member, in this case a printer, that is assigned to the class. Class files are created, modified, and deleted by the **lpadmin**(1M) command. Every class file must always have at least one member.

/etc/lp/interfaces
Contains one executable interface program for each printer that is in the LP Print Service. The filename of the interface program is the same as the printer name. The interface program is invoked with its standard error and standard output directed to the printer. Interface programs may be shell procedures or compiled C programs.

/etc/lp/printers
Contains one directory for each printer. The directory name is the same as the printer name. Each directory contains files defining the printer configuration and other printer specific data.

/usr/lib/lp/model
Contains the printer interface programs that are distributed with the LP Print Service.

/var/spool/lp/requests
Contains one directory for each system on which a printer is attached. The directory names are the same as the system names. Each directory is created at the time a system's printer is first accessed. Upon submitting a print job, the spooler creates a request file and places this file in the appropriate directory. The name of the request file is in the format *seq_num-0*, where *seq_num* is the sequence number assigned to the print request. The file is removed when the job is printed or cancelled.

/var/spool/lp/tmp

Contains one directory for each system on which a printer is attached. The directory names are the same as the system names. Each directory is created at the time a system's printer is first accessed. Upon submitting a print job, the spooler creates a request file and places this file in the appropriate directory. The name of the request file is in the format *seq_num-0*, where *seq_num* is the sequence number assigned to the print request. Along with the request file, other files may appear as well. These files may include the following:

- *Fseq_num-1*
  This file contains the data to be printed after it is processed by a filter.

- *seq_num-1*
  This file contains a copy of the original data file prior to any filtering. This file will only appear if the **–c** option is used when the print job is submitted or the input file is taken from standard input.

These files are removed when the print job is printed or cancelled.

/var/spool/lp/tmp/.net/requests

Contains one directory for the local system on which **lpsched** is running. The directory name is the same as the system name. When a print request is to be sent to a remote system the spooler creates a request file and places the file in the appropriate directory. The request file contains information about the originating system. The name of the request file is the same as the request file created by the spooler and placed in the appropriate directory in */var/spool/lp/requests*. This file is removed when then the print request is sent to the remote system.

/var/spool/lp/tmp/.net/tmp

Contains one directory for the local system on which **lpsched** is running. The directory name is the same as the system name. When a print request is to be sent to a remote system the spooler creates a request file and places the file in the appropriate directory. The name of the request file is the same as the request file created by the spooler and placed in the appropriate directory in */var/spool/lp/tmp*. This file is removed when the requested job is completed on the remote system.

/usr/bin

Contains user commands, administrative commands, filters, and executables invoked by the spooler.

/usr/lib/postscript

Contains postscript template files used by the spooler and postscript related filters.

Lock Files

To guarantee LP Print Service commands exclusive access to data files, several lock files are maintained. They are binary files that contain the process id of the locking process. The lock files and their associated data files are:

- OUTQLOCK — outputq
- PSTATLOCK — pstatus
- QSTATLOCK — qstatus
- SEQLOCK — seqfile

Lock files expire after a given time and may be unlinked by any LP Print Service process. Thus commands that lock a data file for longer than this interval must update the modification time on the lock file. The creating, updating, and unlinking of lock files is handled automatically by the LP Print Service low–level file access routines.

Another lock file, SCHEDLOCK, is present while **lpsched** is running to make sure that only one invocation of **lpsched**(1M) is active. Unlike other lock files, SCHEDLOCK has no expiration time.

Commands for the LP Print Service

The LP Print Service has five user commands and nine administrator commands. The administrator commands require root privileges.

User Commands

These commands enable users to send requests to the LP Print Service, check the status of the requests, and cancel requests. Users can disable and enable a printer, also. Thus a user who finds a malfunctioning printer does not have to call the administrator to turn the printer off.

The user commands, located in the */usr/bin* directory, are shown in Figure 4–1.

Command	Description
enable(1)	Activate the named printer(s).
cancel(1)	Cancel a request for a file to be printed.
disable(1)	Deactivate the named printer(s).
lp(1)	Send a file or files to a printer.
lpstat(1)	Print the status of the LP Print Service.

Figure 4–1: User Commands for the LP Print Service

Administrator Commands

Additional commands located in the */usr/lib* directory are available to the system administrator. Remember, you must have **root** privileges to use these commands.

You should manage your printers through the system administrator menus; however, some options of **lpstat** and **lpadmin** are not available through the menus. To use the administrator commands from the system prompt, you must be logged in as **root**. Figure 4–2 shows the administrator commands.

Command	Description
/usr/lib/accept(1M)	Permit job requests to be queued for a specified destination.
/usr/lib/reject(1M)	Prevent job requests from being queued for a specified destination.
/usr/lib/lpadmin(1M)	Set up or change the LP Print Service configuration. Many features of this command can not be used when **lpsched** is running.
/usr/lib/lpfilter(1M)	Set up or change filter definitions.
/usr/lib/lpforms(1M)	Set up or change pre–printed forms. Use **/usr/lib/lpadmin**(1M) to mount a form.
/usr/lib/lpmove(1M)	Move job requests from one destination to another. This command requires **lpsched** to be running.
/usr/lib/lpsched(1M)	Start the LP scheduler that routes job requests to interface programs that perform the printing on devices.
/usr/lib/lpshut(1M)	Stop the LP scheduler (**lpsched**). See **lpsched**(1M) in the *Reference Manual*.
/usr/lib/lpusers(1M)	Set or change the default priority and priority limits the users of the LP Print Service can request.
/usr/sbin/lpsystem(1M)	Register remote systems with the print service.

Figure 4–2: Administrator Commands for the LP Print Service

Getting Started

Your first tasks are to physically connect your printers to your computers and to install the LP Print Service utilities from the streaming tape on which they were delivered. Once installed, these utilities are available whenever your UNIX system is brought up.

Even after you have installed the hardware and software, however, printers are not available for use immediately. Before users can start submitting requests for print jobs, you must complete the following three steps:

- Configure your printers

  You must name the printers and describe their characteristics to the print service. To configure your printers, use the **lpadmin** command.

- Register your printers

  You must register any printer (or class of printers) that you have configured so that it accepts and queues print requests. To register printers (or classes) run the **accept** command.

- Enable your printers

  You must make your printers active and available to users by running the **enable** command.

Installing the LP Print Service

Procedure

To install the LP Print Service, perform the following steps:

Step 1: Connect your printer (and any optional hardware you may have) to your computer, following the instructions in the appropriate documents.

Step 2: Install the LP Print Service software by entering the following command:

```
# pkgadd -d /dev/rmt/device_name
```

The *device_name* is the device name for the tape drive from which you are installing (for example, *c0t3d0s0*).

The **pkgadd** command prompts you to insert the install media into the device drive at the appropriate time. It also prompts you through the whole installation process.

Controlling Access to the Print Service

During installation, the **pkgadd** command asks you whether you want to authorize the users on your system to enable and disable the printer. You may want to give your users permission to use the **enable** and **disable** commands so that they can turn off a malfunctioning printer without calling the administrator. On the other hand, it may not be reasonable, in your printing environment, to allow regular users to disable printers.

Adding Printers

Why/When

Before the LP Print Service can start accepting print requests, you must describe the characteristics of each printer. Although you can describe your printers extensively, you need to specify only the following information to add a new printer to the LP Print Service.

- To add a local printer, specify the printer name and connection method.
- To add a remote printer, specify the printer name and system name.

To provide more extensive information about your printer, see "Setting Other Printing Options".

Before You Start

- Choose a name for the printer you are adding that is meaningful to the users of the LP Print Service. The printer name identifies the printer, both to you (when you want to change the printer configuration or manage the printer), and to users who want to use the printer. The name may contain a maximum of 14 alphanumeric characters and underscores. For example, **laser** is a good name for a laser printer. If you have several laser printers, you may name them **laser1**, **laser2**, and so on.

  There are no default names; you must name every printer.
- If you are adding a remote printer, be sure you know your system name.

Adding a Local Printer

To connect a printer directly to your computer, enter the following command:

```
# lpadmin -p printer-name -v pathname
```

printer–name is the name of your printer, –v specifies that you are connecting your printer directly, and *pathname* is the name of the special device file representing the printer port.

The following are examples of typical names of special device files:

```
/dev/contty
/dev/term/11
/dev/term/12
/dev/lp
/dev/term/00
```

Using a Printer As a Login Terminal

Some directly connected printers can also be used as terminals for login sessions. If you want to use a printer as a terminal, you must arrange for the LP Print Service to handle it as such.

To do so, use the **–l** option to the **lpadmin** command, as follows:

```
# lpadmin -p printer-name -v path-name -l
```

As before, *path–name* is the name of the special file representing the printer port. If you specify the –l option, the printer is disabled automatically whenever the LP Print Service is started, and you must enable it before it can be used for printing.

Adding a Remote Printer (SVR4)

Use the following procedure to enable users on the local system to access a printer on a remote system running UNIX SVR4. You must perform some of these steps on the local system and others on the remote system.

On the Local System

Step 1: Inform the local system of the remote system with the **lpsystem** command:

```
# lpsystem remote-system
```

Step 2: To add a remote printer, enter the **lpadmin** command at the local system:

```
# lpadmin -p printer -s remote-system
```

Printer is the name by which your users identify the remote printer. You can usually use the same name the remote system uses to identify that printer. If the name used by the remote system is the same name used for an existing printer or class on your system, you must use a different name.

To assign a different name to a remote printer, enter the following:

```
# lpadmin -p lcl-name -s rem-sys!remote-name
```

For example, you want your users to have access to a printer called *psjet2* that resides on a remote system called *newyork*. Because you already have a printer called *psjet2* on your own system, you want to give the remote printer a new name on your system: *psjet3*. Request the new name by entering the following:

```
# lpadmin -p psjet3 -s newyork!psjet2
```

On the Remote System

Step 1: Inform the remote system of the local system with the **lpsystem** command:

```
# lpsystem local_system_name
```

Adding a Remote Printer (BSD)

Use the following procedure to enable users on the local system to access a printer on a remote system running BSD UNIX. You must perform some of these steps on the local system and others on the remote system.

On the Local System

Step 1: Inform the local system of the remote system with the **lpsystem** command:

```
# lpsystem -t bsd bsd-system
```

Step 2: To add a remote printer, enter the **lpadmin** command at the local system:

```
# lpadmin -p printer -s bsd-system
```

Printer is the name by which your users identify the remote printer. You can usually use the same name the remote system uses to identify that printer. If the name used by the remote system is the same name used for an existing printer or class on your system, you must use a different name.

To assign a different name to a remote printer, enter the following:

```
# lpadmin -p lcl-name -s bsd-sys!remote-name
```

For example, you want your users to have access to a printer called *psjet2* that resides on a remote system called *newyork*. Because you already have a printer called *psjet2* on your own system, you want to give the remote printer a new name on your system: *psjet3*. Request the new name by entering the following:

```
# lpadmin -p psjet3 -s newyork!psjet2
```

Step 3: Verify that the local system can communicate with the remote system:

```
# ping bsd-system
```

Step 4: Add both the local and remote remote system names and addresses to the */etc/hosts* file.

Step 5: Create the file */etc/hosts.lpd* and add both the local and remote system names and addresses.

Step 6: Stop the LP Print Service:

```
# /usr/lib/lpshut
```

Step 7: Verify that no **lpNet** or **lpsched** process is running:

```
# ps -ef | grep lp
```

Step 8: Start the **lpsched** process:

```
# /usr/lib/lpsched
```

On the Remote System

Step 1: Inform the remote system of the local system by making the appropriate entry in */etc/hosts.equiv* or */etc/hosts.lpd*.

Step 2: Enter the following command:

```
# /usr/etc/lpc restart all
```

Enabling Remote Printing

To make the printers on your local system accessible to users on remote systems, you must perform the following tasks:

- Prepare the remote system to print on a local printer
- Prepare the local system to accept remote requests

This section provides instructions for these tasks. These instructions are provided specifically for a TCP/IP network and may not work with other networks.

NOTE: The remote print jobs are not visible in the local print queue.

Before You Start

Make sure the network is set up and functioning properly. Use **ping**(1) to make sure each system can communicate with the other.

Prepare the Remote System

You must perform the following tasks on the system where the print job is originating (that is, on the system that does not have the printer attached):

Step 1: Log in as root.

Step 2: Add the following line at the end of the */etc/services* file (if it is not already in the file):

```
listen 2766/tcp
```

If the service name **listen** is already present with another port number and protocol, or if **2766/tcp** is already used for another service, you can not use this example without changes. This example assumes that the **listen** service has the port number and protocol **2766/tcp**.

Step 3: Stop the print service:

```
# lpshut
```

Step 4: Define the server that has the printer connected. If the server with the printer is running System V UNIX, enter the following command:

```
# lpsystem local_server_name
```

If the server with the printer is running BSD UNIX, enter the following command:

```
# lpsystem -tbsd local_server_name
```

The *local_server_name* is the name of a server (with a printer) on the network which has a defined internet address in the */etc/hosts* file. Make sure the system name used is the same as the system name used when the printer was configured.

Step 5: Start the LP Print Service:

```
# lpsched
```

Step 6: Set up the printer definition using:

```
# lpadmin -p printer -T type -s local_server
```

The *type* parameter refers to the type of printer, such as *hppaintjet*.

Step 7: Accept print requests by performing the following command:

```
# accept printer_name
```

Step 8: Enable the printer with the following command:

```
# enable printer_name
```

Prepare the Local System

You must perform the following tasks on the system where the print job is being sent (that is, on the system where the printer is attached):

Step 1: Login as root.

Step 2: Set up the printer definition for the printer connected to the parallel port. You may use the **sysadm** menus or the following commands:

```
# chmod 600 /dev/lp
# chown lp /dev/lp
# chgrp bin /dev/lp
# lpadmin -p printer_name -T type -v /dev/lp
# accept printer_name
# enable printer_name
```

NOTE: You may use a temporary file such as */tmp/filelp* (*filelp* could be any file name) in place of */dev/lp*.

The *printer_name* and *type* should be the same as those entered on the remote system in Step 6 of the procedure "At the Remote System." If the printer is connected to a device other than the parallel port (for example the serial port or a terminal server), the device name */dev/lp* must be replaced accordingly.

Step 3: Stop the print service by entering the following command:

```
# lpshut
```

Step 4: Define the remote server (without the printer) to the local system. If the local system (without the printer) is running System V UNIX, enter the following command:

```
# lpsystem remote_server_name
```

If the local system (without the printer) is running BSD UNIX, make the necessary entries in */etc/host* and */etc/hosts.equiv* so that the remote system can communicate with this local system.

Step 5: Start the print service by entering the following command:

```
# /usr/lib/lp/lpsched
```

Step 6: Initialize the listener process by entering the following command:

```
# nlsadmin -i tcp
```

If this command has been performed before, an error message informs you that the listener has already been initialized. Ignore the message.

Step 7: Determine the TCP/IP hexadecimal address of the local server (with the printer). If the remote server (without the printer) is running System V UNIX, enter the following command:

```
# nlsadmin -l '/usr/etc/rfsaddr -h \
remote_sersver_name' tcp
```

If the remote server (without the printer) is running BSD UNIX, enter the following command:

```
# /usr/etc/rfsaddr -h remote_server_name
```

Step 8: Configure the port monitor so the network can accept service requests for remote printing. If the remote server (without the printer) is running System V UNIX, enter the following command:

```
# pmadm -a -p tcp -s lp -i root -v 'nlsadmin \
-V' -m 'nlsadmin -o \
/var/spool/lp/fifos/listenS5'
```

If this command has previously been performed, an error message informs you that **lp** already exists under tcp. If this occurs, perform the **pmadm –r –p tcp –s lp** command and then perform Step 8 again.

If the remote system (without the printer) is running BSD UNIX, enter the following command:

```
# pmadm -a -p tcp -s lpd -i root -v \
'nlsadmin -V' -m 'nlsadmin -o \
/var/spool/lp/fifos/listenBSD -A \
"\x00020203xaddress"'
```

The *xaddress* represents the last eight digits of the hexadecimal address determined in Step 7. The *0203* part of the address represents port 515.

Troubleshooting

If remote printing does not work after you have performed the previous tasks, try the following:

Step 1: Ensure that the address for the listen process on the receiving machine matches the address being sent in the print request.

Run the **pmadm –l** command on the machine with the printer. The output is a hex address associated with the NLPS server **listen** process (the actual address is found in the last 8 digits).

This address should match the address returned when **/usr/etc/rfsaddr –h** *name* is run on the machine without the printer, where *name* is the name of the machine with the printer. If they do not match, you must change one of the values.

- Change the */etc/hosts* file or Domain Name Services database.
- Change the address associated with the NLPS server, use the **nlsadmin –l** command.

You must decide which one is in error and correct the problem.

Step 2: Ensure that the port number for the listen process on the machine with the printer matches the port number being sent in the print request. These may not match when you are setting up remote printing between a system running Release 2.xx of NCR UNIX SVR4 and another running Release 1.xx.

The */etc/services* file on the machine without the printer should have an entry for **listen** (as defined in setting up the machine without the printer). The number associated with this entry is the port number being used in the print request. If you convert this number to hex, it should match the 4 digits preceding the address in the **pmadm –l** command.

The most common values and their hex representations are:

```
1025  0401
2766  0ace
```

To bring these numbers into sync, either change the */etc/services* file on the local system or enter the following command:

```
# nlsadmin -l correct_value tcp
```

To find the *correct_vlaue*, enter the following command:

```
# /usr/etc/rfsaddr -h remote_name
```

The port number (which must be written in hex) consists of the four hex digits following the *x0002* characters.

Changing the port on the remote machine may affect other communication with it, such as **uucp** and **lp**. The recommended port is **2766**. The most common other value is **1025**.

If the remote system uses port 1025 and you want to change it to 2766, change the port for **uucp** as well. Edit the */usr/lib/uucp/Systems* file and change the value *\x0020401* to *\x0020ace*.

If remote printing was previously set up, edit the */etc/services* file and change the *listen 1025/tcp* entry to *listen 2766/tcp*.

These changes may cause listeners on other machines to have their port number changed, causing other system to have to change their */etc/services* and **uucp** files. Therefore, all systems in the network should use the same port number for the listener process.

If you use multiple port numbers on your network you must resolve the following issues:

- For any server, all other systems trying to reach it for LP must have the same port number in their */etc/services* file.
- For any server, all other systems trying to reach it for **uucp** over TCP/IP must have the same port number in the */usr/lib/uucp/Systems* file.
- For any system trying to remote print to more than one server, all destination servers must be listening on the same port, which must match the value in the */etc/services* file for the system requesting to print.
- For any system trying to reach other systems for **uucp**, each entry in the */usr/lib/uucp/Systems* file may have a different port number. However, the port number must match the port number used by the listener on the corresponding system.

Managing Printer Classes

Treating a collection of printers as a single class allows users to submit files for printing by a member of a class. The LP Print Service prints the job on the first printer in the class that it finds free. This method allows faster turn–around, as printers are kept as busy as possible.

This section contains procedures for the following tasks:

- Adding printer classes
- Removing printers or classes

Add Printer Classes

NOTE: This section does not apply if you are making a remote printer accessible to users on your system.

You might define printer classes in order to:

- Specify a series of printers that should be used in a particular order
- Group printers that can print specialized files, using print wheels, forms, or character sets
- Group printers that are geographically close to each other

To add a printer to a class, use the following command:

```
# lpadmin -p printer-name -c class-name
```

If the class *class–name* does not exist yet, it is created.

NOTE: Class names and printer names must be unique. Because they are unique, a user can specify the destination for a print request without having to know whether it is a class of printers or a single printer.

Remove a Printer or Class

You can remove a printer or class if it has no pending print requests. If there are pending requests, you have to first move them to another printer or class (using the **lpmove** command), or cancel them (using the **cancel** command).

Removing the last remaining printer of a class automatically removes the class as well. Removing a class, however, does not remove printers that were members of the class. If the printer or class removed is also the system default destination, the system no longer has a default destination.

To remove a printer or class, enter the following command:

```
# lpadmin -x printer-or-class-name
```

If you want to remove a printer from a class without deleting that printer, enter the following command:

```
# lpadmin -p printer-name -r class-name
```

Setting Other Printing Options

After you have added printers to your system, you may also want to specify these optional pieces of information:

- Printer type
- Content type
- Printer port characteristics
- Banner page
- Printer description
- Default printing attributes
- Printer destinations

This section contains procedures for specifying this information.

Specify a Printer Type

A printer type is the generic name for a printer. Typically it is derived from the manufacturer's name, such as **572** for the AT&T 572 Dot Matrix Printer. When you set up your system, you can enhance its ability to serve your users by classifying, on the basis of type, the printers available through the print service.

If your printer is capable of emulating more than one kind of printer, you can assign several types to it. If you specify more than one printer type, the LP Print Service uses one of them as appropriate for each print request.

While you are not required to specify a printer type, we recommend that you do so; when you specify a printer type, better print services can be provided.

To specify a printer type, use the following command line:

```
# lpadmin -p printer-name -T printer-type-list
```

If you give a list of printer types, separate the names with commas. If you do not define a printer type, the default "unknown" is used.

Specify Content Types

The content types tell the LP Print Service what types of files can be printed directly on each printer. Most printers can print files of two types: the same type as the printer type (if the printer type is defined) and the type **simple**, (meaning an ASCII file) which is the default content type for all printers.

Some printers, though, can accept (and print properly) several different types of files. When adding this kind of printer, specify the names of the content types the new printer accepts by adding these names to the list. (By default, the list contains only one type: **simple**.) If you are adding a remote printer, list the content types that have been established for it by the administrator of the system on which it resides.

Procedure

To specify the list of content types, enter the following command:

```
# lpadmin -p printer-name -I content-type-list
```

The *content-type-list* is a list of names separated by commas or spaces. If you use spaces to separate the names, enclose the entire list (but not the **-I**) in quotes.

Additional Information

Content type names may look a lot like printer type names, but you are free to choose names that are meaningful to you and the people using the printer. Remember the following restrictions:

- The names **simple** and **any** have particular meanings to the LP Print Service; use them consistently.
- The name **terminfo** is reserved as a reference to all types of printers.
- The names must contain no more than 14 characters and may include only letters, digits, and underscores.

Figure 4–3 lists and describes some accepted content types.

Types	Description
troff	Device independent output from **troff**
otroff	CAT typesetter instructions generated by BSD or pre–System V **troff** (old **troff**)
tex	DVI format files
plot	Plotting instructions for Tektronix displays and devices
raster	Raster bitmap format for Varian raster devices
cif	Output of BSD **cifpbt**
fortran	ASA carriage control format
postscript	PostScript language
simple	ASCII file

Figure 4–3: Accepted content types

If the same content type is printable by several different types of printers, you should use the same content type names when you add those printers. Users can then use the same name to identify the type of file they want printed regardless of the printing destination.

If you do not list the content types for each printer, the printer type is used as the name of the content type the printer can handle. If you have not specified a printer type, the LP Print Service assumes the printer can print only files of content type **simple**. This may be sufficient if you require users to specify the proper printer explicitly and if files are properly prepared for the printer before being submitted for printing.

The Default Content Type: simple

Files of content type **simple** are assumed to contain only two types of characters, printable ASCII characters and the following control characters:

backspace
: Moves the carriage back one space, except at the beginning of a line

tab
: Moves the carriage to the next tab stop; by default, stops are spaced every 8 columns on most printers

linefeed
: Moves the carriage to the beginning of the next line (may require special port settings for some printers–see "Specifying Printer Port Characteristics")

form feed
: Moves the carriage to the beginning of the next page

carriage return
: Moves the carriage to the beginning of the same line (may fail on some printers)

If a printer can handle several types of files, including **simple**, you must include **simple** in the content type list; the type **simple** is not automatically added to any list you give. If you do not want a printer to accept files of type **simple**, give a blank *content–type–list*, as follows:

```
# lpadmin -p printer-name -I ""
```

Specify Printer Port Characteristics

NOTE: This section does not apply if you are making a remote printer available to users on your system.

Printers connected directly to computers and those connected over some networks require that the printer port characteristics be set by the interface program. The standard interface program uses the **stty** command to initialize the printer port, minimally setting the following default characteristics shown in Figure 4–4.

Default	Meaning
9600	9600 baud rate
cs8	8–bit bytes
–cstopb	1 stop bit per byte
–parenb	no parity generation
ixon	enable XON/XOFF flow control
–ixany	allow only XON to restart output
opost	post–process data stream as listed below:
–olcuc	don't map lower–case to upper–case
onlcr	map linefeed into carriage–return/linefeed
–ocrnl	don't map carriage–return into linefeed
–onocr	output carriage–returns even at column 0
nl0	no delay after linefeeds
cr0	no delay after carriage–returns
tab0	no delay after tabs
bs0	no delay after backspaces
vt0	no delay after vertical tabs
ff0	no delay after form–feeds

Figure 4–4: Default Printer Port Characteristics

These default characteristics may be sufficient for your printers. However, printers vary enough that you may have to set different characteristics. See the description of the stty(1M) command in the *Reference Manual* to find the complete list of characteristics.

If you have a printer that requires printer port characteristics other than those handled by the stty program, you must customize the interface program. See the section "Customizing the Print Service" in the "Adjusting the LP Print Service" chapter for help.

Procedure

You may specify an additional list of port characteristics. The list you give is applied after the default list, so that you do not need to include in your list items that you do not want to change.

Specify the additional list as follows:

```
# lpadmin -p printer-name -o "stty='stty-option-list'"
```

Note that both the double quotes and single quotes are needed if you give more than one item in the *stty-option-list*.

Examples

Your printer is to be used for printing graphical data, where linefeed characters should be output alone, without an added carriage-return. Enter the following command:

```
# lpadmin -p printer-name -o "stty=-onlcr"
```

Note that the single quotes are omitted because there is only one item in the list.

Suppose your printer requires odd parity for data sent to it. You would enter the following command:

```
# lpadmin -p printer-name -o "stty='parenb \
parodd cs7'"
```

Specify a Banner Page in Output

Most users want to have the output of each print request preceded by a banner page. A banner page shows who requested the printing, the request ID for it, and when the output was printed. It also allows for an optional title that the requester can use to better identify a printout. Finally, the banner page greatly eases the task of separating a sequence of print requests so that each may be given to the correct user.

Sometimes a user needs to avoid printing a banner page (for example, when the printer has forms mounted that should not be wasted, such as payroll checks or accounts payable checks). Printing a banner page under such circumstances may cause problems.

If you do not allow a user to skip the banner page, the LP Print Service rejects all attempts to avoid a banner page when printing on the printer (the default action).

Enter the following command to allow users to request no banner page:

```
# lpadmin -p printer-name -o nobanner
```

If you later change your mind, you can reverse this choice by entering the following command:

```
# lpadmin -p printer-name -o banner
```

Describe Your Printer

To give users of the LP Print Service helpful information about a printer, add a description of it. This description can contain any message, including the number of the room where the printer is found, the name of the person to call with printer problems, and so forth.

To add a description of a printer, enter the following command.

```
# lpadmin -p printer-name -D 'text'
```

The *text* is the message. Include the quotes if the message contains blanks or other characters that the shell might interpret if the quotes are left out.

Change Default Printing Attributes

The attributes of a printing job include the page size, print spacing (character pitch and line pitch), and **stty** options for that job. If a user requests a job to be printed on a particular form, the printing attributes defined for that form are used for that job. However, when a user submits a print request without requesting a form, the print service uses one of several sets of default attributes.

The LP Print Service lets you override the defaults for each printer, allowing you to designate different printers as having different default page sizes (width and length) or print spacing (character and line). A user can then simply route a file to the appropriate printer to get a desired style of output. For example, you can have one printer dedicated to printing wide (132–column) output, another printing normal (80–column by 66–line) output, and yet another printing letter quality (12 characters per inch, 8 lines per inch) output.

Set the defaults using one or more of the following commands:

```
# lpadmin -p printer-name -o width=scaled-number
# lpadmin -p printer-name -o length=scaled-number
# lpadmin -p printer-name -o cpi=scaled-number
# lpadmin -p printer-name -o lpi=scaled-number
```

Append the letter **i** to the *scaled–number* to indicate inches, or the letter **c** to indicate centimeters. The letter **i** for character pitch (cpi) or line pitch (lpi) is redundant. You can also give **pica**, **elite**, or **compressed** instead of a number for the character pitch.

Define Default Destinations

You can define the printer or class to be used to print a file when the user has not explicitly asked for a particular destination and has not set the **LPDEST** shell variable.

Before You Start

The printer or class must already exist.

Procedure

To make a printer or class the default destination, enter the following command:

```
# lpadmin -d printer-or-class-name
```

If you later decide that there should be no default destination, enter a null *printer–or–class–name* as in the following command:

```
# lpadmin -d
```

If you do not set a default destination, users must explicitly name a printer or class in each print request, (unless they specify the –T *content–type* option) or set the **LPDEST** shell variable with the name of a destination.

Putting It All Together

You can set printer options one at a time, or you can combine several on one command line. Below are some examples.

Example 1

Add a new printer called **lp1** (of the type 455) on printer port **/dev/term/13**. It should use the standard interface program, with the default page size of 90 columns by 71 lines, and linefeeds should *not* be mapped into carriage return/linefeed pairs. (The following command line is split into multiple lines for readability.)

```
# lpadmin -p lp1 -v /dev/term/13 -T 455 -o \
"width=90 length=71 stty=-onlcr"
```

Example 2

Add a new printer called **laser** on printer port **/dev/term/41**. It should use a customized interface program, located in the directory */usr/doceng/laser_intface*. It can handle three file types—i10, i300, and impress—and it may be used only by the users doceng and docpub. (The following command line is split into multiple lines for readability.)

```
# lpadmin -p laser -v /dev/term/41 -i \
/usr/doceng/laser_intface -I "i10,i300,impress" \
-u "allow:doceng,docpub"
```

Displaying the Status of Printers

Why/When

Use the **lpstat** command to examine both the configuration and the current status of a printer. The short form of this command gives just the status; you can use it to see if the printer exists and if it is busy, idle, or disabled. The long form of the command gives a complete configuration listing.

After defining a printer configuration, you may want to review it to see if it is correct. If you find you have made a mistake, just reenter the command that applies to the part that is wrong.

Procedure

Enter the following command to examine a printer:

```
# lpstat -p printer-name -l
```

You should see one of the following lines of output:

```
printer printer-name now printing request-id. \
enabled since date.

printer printer-name is idle. enabled since date.

printer printer-name disabled since date.
reason

printer printer-name waiting for auto-retry.
reason
```

The "waiting for auto–retry" output shows that the LP Print Service failed in trying to use the printer (because of the reason shown), and that it will try again later.

You should also see the following output:

```
Form mounted: form-name
Content types: content-type-list
Printer type: printer-type
Description: comment
Connection: connection-info
Interface: path-name
On fault: alert-method
After fault: fault-recovery
Users allowed:
   user-list
Forms allowed:
   form-list
Banner required
Character sets:
   character-set-list
Default pitch: integer CPI, integer LPI
Default page size: paperwidth wide,
   paperlengthlong
Default port settings: stty-option-list
```

Starting and Stopping the LP Print Service

Under normal operation, you should not have to start or stop the LP Print Service manually. It is automatically started each time the UNIX system is started, and stopped each time the UNIX system is stopped. If, however, you need to stop the LP Print Service without stopping the UNIX system as well, you can do so by following the procedure described below.

Stopping the LP Print Service causes all printing to cease within seconds. Any print requests that have not finished printing are printed in their entirety after the LP Print Service is restarted. The printer configurations, forms, and filters in effect when the LP Print Service is stopped are restored after it is restarted.

NOTE: To start and stop the LP Print Service manually, you must be logged in as either the user **lp** or the superuser (**root**).

Start the Print Service

Procedure

To start the LP Print Service manually, enter the following command:

```
# /usr/lib/lpsched
```

Additional Information

The following message appears:

```
Print services started.
```

It may take a minute or two for the printer configurations, forms, and filters to be re-established before any saved print requests start printing.

If you try to restart the LP Print Service when it is already running, you receive the following message:

```
Print services already active.
```

NOTE: You do not have to stop the LP Print Service to change printer configurations or to add forms or filters.

Stop the Print Service

Procedure

To stop the LP Print Service manually, enter the following command:

```
# lpshut
```

Additional Information

The following message appears:

```
Print services stopped.
```

All printing ceases within a few seconds.

If you try to stop the LP Print Service when it is not running, you see the following message:

```
Print services already stopped.
```

Managing the Printing Load

To manage the operation of printers on your system efficiently, you may need to take printers in and out of service for repairs or to adjust the load on individual printers.

The procedures in this section allow you to:

- Accept requests for a printer or class
- Reject requests for a printer or class
- Enable a printer
- Disable a printer
- Give users permission to disable and enable printers
- Move print jobs to another destination

Accept Requests for a Printer or Class

You must use the **accept** command for a new printer or class after you have added it, because the LP Print Service does not initially accept requests for new printers or classes.

Enter one of the the following commands to start accepting new requests.

```
# accept printer-name
# accept class-name
```

You can accept requests for several printers or classes in one command by listing their names on the same line.

Reject Requests for a Printer or Class

You may need to stop **lp** from routing requests to a specific printer. For example, if a printer has been removed for repairs or if too many requests are queued to this destination, you may want to prevent new jobs from being sent to it.

Although the **reject** command stops any new print requests from being accepted, it does not move or cancel any requests currently queued for the printer. These are printed as long as the printer is enabled.

To stop accepting any new requests for a printer or class of printers, enter the following command:

```
# reject -r "reason" printer-or-class-name
```

You can reject requests for several printers or classes in one command by listing their names on the same line, separating the names with spaces. The *reason* is displayed whenever anyone tries to print a file on the printer. You can omit it (and the **–r**) if you don't want to specify a reason.

Enable a Printer

The LP Print Service waits for an explicit signal from you before it starts printing files. Once you have verified that all the necessary components of your printer (print wheels, forms, and so on) are in place, you can request the beginning of printing by issuing the enable command.

Enter the following command:

```
# enable printer-name
```

To enable several printers simultaneously, list the printers (separating the names with spaces) on the same line as the **enable** command. Do not enclose the list in quotes.

Disable a Printer

Disabling a printer stops further print requests from being printed. (It does not, however, stop the LP Print Service from accepting new print requests for the printer.) From time to time you may want to disable a printer. For example, you may want to interrupt a print request, or you may want to change a form or print wheel, in which case you should disable the printer first.

Normally, disabling a printer also stops the request that is currently printing, placing it back in the queue so it can be printed later. You can, however, have the LP Print Service wait until the current request finishes, or even cancel the request outright.

To disable a printer, enter one of the following commands:

```
# disable -r "reason" printer-name
# disable -W -r "reason" printer-name
# disable -c -r "reason" printer-name
```

The first command disables the printer, stopping the currently printing request and saving it for printing later. The other commands also disable the printer, but the second one makes the LP Print Service wait for the current request to finish, while the third cancels the current request.

NOTE: The **–c** and **–W** options are not valid when the **disable** command is run to stop a remote printer because, when run for a remote printer, **disable** stops the transferring (rather than the actual printing) of print requests.

The *reason* is stored and displayed whenever anyone checks the status of the printer. You can omit it (and the **-r** option) if you do not want to specify a reason.

To disable several printers at once, list their names in the same line as the **disable** command.

NOTE: You can only enable or disable local printers. You can, however, enable or disable the transfer of print requests to the remote system on which a printer is located. Only individual printers can be enabled and disabled; classes can not.

Allow Users to Enable and Disable a Printer

You may want to make the **enable** and **disable** commands available to other users. This availability is useful, for instance, if you have a small organization where anyone who spots a problem with the printer should be able to disable it and solve the problem. This is NOT a good idea if you want to keep others from interfering with the proper operation of the print services.

To allow everybody to run **enable** and **disable**, enter the following two commands:

```
# chown lp /usr/bin/enable /usr/bin/disable
# chmod u+s /usr/bin/enable /usr/bin/disable
```

The first command makes the user **lp** the owner of the commands; this step should be redundant, but it is safer to run the command than to skip it.

The second command turns on the setuid bit. (Clearing the bit removes the user permission to enable and disable the printer.)

To prevent others from running **enable** and **disable**, enter the following command:

```
# chmod u-s /usr/bin/enable /usr/bin/disable
```

Move Requests to Another Printer

You may need to move output requests between destinations. For example, when a printer is down for repairs, you can move its pending requests to a working printer.

To move requests from one printer or class to another, enter one of the following commands:

```
# lpmove request-id(s) printer-name
# lpmove printer-name1 printer-name2
```

The first command moves the listed requests to the printer *printer–name*.

The second command tries to move *all* requests currently queued for *printer-name1* to *printer-name2*. When the second command is used, the LP Print Service also stops accepting requests for *printer-name1*.

If some requests can not be printed on the new printer, they are left in the queue for the original printer.

Restrict User Access

You can control which users are allowed to use a particular printer on your system. For instance, if you are designating one printer to handle sensitive information, you do not want all users to be able to use the printer. You also might want to restrict access when you are authorizing the use of a high quality printer which produces expensive output.

If you do not use this option, the LP Print Service assumes that everybody may use the printer.

Additional Information

The LP Print Service uses a list of users allowed or denied access to a printer. The LP Print Service rejects a user's request to print a file on a printer he or she is not allowed to use.

If your users have access to remote printers, or if users on other systems have access to printers on your system, make sure that the allow and deny lists for those printers on your computer match the allow and deny lists on the system where the remote printers reside.

NOTE: If these two sets of lists do not match, your users may receive conflicting messages (some accepting jobs, and others refusing jobs) when submitting requests to remote printers.

The method of listing the users allowed or denied access to a printer is similar to the method used to list users allowed or denied access to the **cron** and **at** facilities. Briefly, the rules are as follows:

- An allow list is a list of users allowed to use the printer. A deny list is a list of users denied access to the printer
- If the allow list is not empty, only the users listed are allowed; the deny list is ignored. If the allow list is empty users listed in the deny list are not allowed. If both lists are empty there are no restrictions on who may use the printer
- Specifying all in the allow list allows everybody access to the printer; specifying all in the deny list denies access to everybody except the user **lp** and the superuser (**root**).

Procedure

You can add names of users to either list using one of the following commands:

```
# lpadmin -p printer-name -u allow:user-list
# lpadmin -p printer-name -u deny:user-list
```

The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list.

The *user–list* is a comma or space separated list of user names. If you use spaces to separate the names, enclose the entire list (including the **allow:** or **deny:** but not the **–u**) in quotes. Each item in the *user–list* may take any of the following forms:

user

User on any system

all

All users on all systems

local–system!user

User on *local–system* only

!*user*
User on local system only

all!*user*
User on any system

all!all
All users on all systems

system!all
All users on *system*

!all
All users on local system

Examples

Here are some examples of how you might use these commands:

Example 1

It is time to change the ribbon and perform some preventive maintenance on printer lp1. First, to prevent the loss of print requests already queued for lp1, you move all requests from printer lp1 to printer lp2.

```
# lpmove lp1 lp2
```

After the requests are moved, make sure the LP Print Service does not print any more requests on lp1 by disabling it.

```
# disable lp1
```

Now you may physically disable the printer and start working on it.

Example 2

You have finished changing the ribbon and doing the other work on lp1; now it is time to bring it back into service. Execute the following commands in any order:

```
# accept lp1
# enable lp1
```

Example 3

You notice that someone has queued several large files for printing on the printer laser1. Meanwhile laser2 is idle because no one has queued requests for it. Move the two biggest requests (laser1–23 and laser1–46) to laser2, and reject any new requests for laser1 for the time being.

```
# lpmove laser1-23 laser1-46 laser2
# reject -r "too busy-- will reopen later" \
laser1
```

Troubleshooting

Here are a few suggestions of what to do if you are having difficulty getting a printer to work.

No Output (Nothing Is Printed)

The printer is sitting idle; nothing happens. First, check the documentation that came with the printer to see if there is a self–test feature you can invoke; make sure the printer is working before continuing.

There are three possible explanations when you do not receive any output:

- The printer might not be connected to the computer
- The printer might not be enabled
- The baud rate for the computer and the printer might not be set correctly

The rest of this section describes each of these situations in detail.

Is the Printer Connected to the Computer?

The type of connection between a computer and a printer may vary. The *NCR 345x/35xx Unit Installation, Care and Cleaning* manual provides detailed instructions for connecting printers.

Is the Printer Enabled?

The printer must be "enabled" in two ways: First, the printer must be turned on and ready to receive data from the computer. Second, the LP Print Service must be ready to use the printer. If you receive error messages when setting up your printer, follow the "fixes" suggested in the messages. When the printer is set up, issue the following commands:

```
# accept printer-name
# enable printer-name
```

printer–name is the name you assigned to the printer for the LP Print Service. Now submit a sample file for printing:

```
# lp -d printer-name file-name
```

Is the Baud Rate Correct?

If the baud rate (the rate at which data is transmitted) is not the same for both the computer and the printer, sometimes nothing prints (see "Is the Baud Rate Correct?" in the next section).

Illegible Output

The printer tries printing, but the output is not what you expected; it certainly isn't readable. There are five possible explanations for this situation:

- The baud rate for the printer might not match the baud rate for the computer
- The parity setting of the computer might be incorrect
- The tabs might be set incorrectly
- The printer type might be set incorrectly

The rest of this section describes each of these situations in detail.

Is the Baud Rate Correct?

Usually, when the baud rate of the computer does not match that of the printer, you get some output but it does not look at all like what you submitted for printing. Random characters appear, with an unusual mixture of special characters and unlikely spacing.

Read the documentation that came with the printer to find out what its baud rate is. It should probably be set at 9600 baud for optimum performance. If it is not set to 9600 baud, you can have the LP Print Service use the correct baud rate (by default it uses 9600).

NOTE: If the printer is connected via a parallel port, the baud rate is irrelevant.

To set a different baud rate for the LP Print Service, enter the following command:

```
# lpadmin -p printer-name -o stty=baud-rate
```

Now submit a sample file for printing.

Is the Parity Setting Correct?

Some printers use a "parity bit" to ensure that the data received for printing has not been garbled in transmission. The parity bit can be encoded in several ways; the computer and the printer must agree on which one to use. If they do not agree, some characters either are not printed or are replaced by other characters. Generally, though, the output looks approximately correct, with the spacing of "words" typical for your document and many letters in their correct place.

Check the documentation for the printer to see what the printer expects. The LP Print Service does not set the parity bit by default. You can change this, however, by entering one of the following commands:

```
# lpadmin -p printer-name -o stty=oddp
# lpadmin -p printer-name -o stty=evenp
# lpadmin -p printer-name -o stty=-parity
```

The first command sets odd parity generation, the second sets even parity. The last command sets the default, no parity.

If you are also setting a baud rate other than 9600, you may combine the baud rate setting with the parity settings, as in the sample command below.

```
# lpadmin -p printer-name -o "stty='evenp 1200'"
```

Tabs Set Correctly?

If the printer does not expect to receive tab characters, the output may contain the complete content of the file, but the text may appear in a chaotic looking format, jammed up against the right margin.

Legible Printing, but Wrong Spacing

The output contains all of the expected text and may be readable, but the text appears in an undesirable format: double spaced, with no left margin, run together, or out of alignment down the page. These problems can be fixed by adjusting the printer settings (if possible) or by having the LP Print Service use settings that match those of the printer. The rest of this section provides details about solving each of these types of problems.

Double Spaced

Either the printer's tab settings are wrong or the printer is adding a linefeed after each carriage return. (The LP Print Service has a carriage return added to each linefeed, so the combination causes two linefeeds.) You can have the LP Print Service not send tabs or not add a carriage return by using the stty –tabs option or the –onlcr option, respectively.)

```
# lpadmin -p printer-name -o stty=-tabs
# lpadmin -p printer-name -o stty=-onlcr
```

No Left Margin/Runs Together/Jammed Up

The printer's tab settings are not correct; they should be set every 8 spaces. You can have the LP Print Service not send tabs by using the –tabs option.

```
# lpadmin -p printer-name -o stty=-tabs
```

Out of Alignment Down the Page

The *stty onlcr* option is not set. This option is set by default, but you may have cleared it accidentally. To reset this option, enter the following command:

```
# lpadmin -p printer-name -o stty=onlcr
```

A Combination of Problems

If you need to use several of these options to take care of multiple problems, you can combine them in one list, as shown in the sample command below. Include any baud rate or parity settings, too.

```
# lpadmin -p printer-name-o "stty='-onlcr -tabs 2400'"
```

Wrong Character Set or Font

If you selected the wrong printer type when you set up the printer with the LP Print Service, the wrong "control characters" can be sent to the printer. The results are unpredictable and may cause output to disappear or to be illegible, making it look like a problem described above. Another result may be that the wrong control characters cause the printer to set the wrong character set or font.

If you do not know which printer type to specify, try the following to examine the available printer types. First, if you think the printer type has a certain name, try the following command.

```
# tput -T printer-type longname
```

The output of this command appears on your terminal: a short description of the printer identified by the printer–type. Try the names you think might be right until you find one that identifies your printer.

If you do not know what names to try, you can examine the **terminfo** directory to see what names are available. Enter the following command to examine the directory:

```
# ls -R /usr/share/lib/terminfo/*
```

Pick names from the list that match one word or number identifying your printer. For example, the name **495** would identify the AT&T 495 Printer. Try each of the names in the **tput** command shown earlier.

When you have the name of a printer type you think is correct, set it in the LP Print Service by entering the following command:

 # lpadmin -p *printer-name* -T *printer-type*

Dial Out Failures

The LP Print Service uses the Basic Networking Utilities to handle dial out printers. If a dialing failure occurs and you are receiving printer fault alerts, the LP Print Service reports the same error reported by the Basic Networking software for similar problems. (If you have not arranged to receive fault alerts, they are mailed, by default, to the user **lp**.)

Idle Printers

There are several reasons why you may find a printer idle and enabled but with print requests still queued for it:

- If the device name for this printer is the same as the device name of another printer which is currently active, or in case of network printers, if the dial token is the same as another printer which is currently busy, the printer status is idle and enabled, but the queued request is not being printed.

- The print requests need to be filtered. Slow filters run one at a time to avoid overloading the system. Until a print request is filtered (if it needs slow filtering), it does not print.

  Use the following command to see if the first waiting request is being filtered:

  ```
  # lpstat -o -l
  ```

- The printer has a fault. After a fault is detected, printing resumes automatically, but not immediately. The LP Print Service waits about one minute before trying again, and continues trying until a request is printed successfully. You can force a retry immediately by enabling the printer as follows:

  ```
  # enable printer-name
  ```

- A dial out printer is busy or does not answer, or all dial out ports are busy. As with automatic continuation after a fault, the LP Print Service waits one minute before trying to reach a dial out printer again. If the dial out printer can not be reached for an hour or two (depending on the reason), the LP Print Service finally alerts you to a possible problem. You can force a retry immediately by enabling the printer as follows:

  ```
  # enable printer-name
  ```

Networking Problems

You may encounter several types of problems while trying to get files printed over a network:

- Requests being sent to remote printers may back up in the local queue
- Requests sent to remote printers may back up in the remote queue
- A user may receive contradictory messages about whether a remote printer has accepted a print request

The rest of this section describes each of these situations and suggests how to resolve them.

Jobs Backing Up in the Local Queue

Several jobs are backing up in the local queue for a remote printer. There are three possible explanations:

- The remote system is down or the network between the local and remote systems is down. To resolve this problem, run the **reject** command for all the remote printers on your system, as follows:

  ```
  # reject printer-name
  ```

  This command stops new requests for those printers from being added to the queue. Once the system comes up again and jobs start being taken from your queue, type **accept** *printer* to allow new jobs to be queued.

- The remote printer is disabled on the local system.
- The underlying System V network software is not set up properly. For details, see **lpsystem**(1M).

Jobs Backing Up in the Remote Queue

The remote printer has been disabled.

Conflicting Messages About the Acceptance/Rejection of Jobs

A user enters a print request and is notified that the system has accepted it. The job is sent to a remote system and the user receives mail that the job has been rejected.

The definitions, on the local and remote systems, of print job components (such as filters, character sets, print wheels, and forms) do not coincide. Identical definitions of these job components must be registered on both the local and the remote systems if local users are to be able to access remote printers.

This problem may occur for one of two reasons:

- The local computer may be accepting requests while the remote computer is rejecting requests

- The definitions of the remote printer on the local computer may not match the definition of that printer on the remote computer. As a result, the scheduler is gone and has no record of the removed job. Therefore the user is not notified that the job has been removed.

Chapter 5

Managing the LP Print Service

This chapter contains procedures for more advanced printer maintenance. It includes the following tasks and information:

Managing Queue Priorities

The LP Print Service provides a priority mechanism that allows users to adjust the position of a print request in the queue. Each print request can be given a priority level by the user who submits it; a priority level is a number from 0 to 39, with *smaller* numbers indicating *higher* levels of priority. Requests with higher priority (smaller numbers) are placed ahead of requests with lower priority (larger numbers).

A priority scheme this simple would not work if there were no controls on how high users can set the priority. You can define the following characteristics in this scheme:

- A priority limit for each user. Users can not submit a print request with a priority higher than their limit, although they can submit a request with a lower priority.
- A default priority limit for users who are not assigned a personal limit.
- A default priority for print requests to which the user does not assign a priority.

By setting the characteristics according to your needs, you can prevent lower priority printing tasks (such as regular printing by most staff members) from interfering with higher priority printing tasks (such as payroll check printing by the accounting staff).

You may want a critical print request to print ahead of any others, perhaps even if it has to preempt the currently printing request. You can have the LP Print Service give immediate handling to a print request, and you can have it put another print request on hold. This prints the first request and delays the second print request until you allow it to be resumed.

This section describes the following tasks:

- Setting Priority Limits
- Setting a Default Priority
- Examining the Priority Limits and Defaults
- Moving a Request Around in the Queue

Set Priority Limits

For A Specific User

To set a user's priority limit, enter the following command:

```
# lpusers -q priority-level -u user-name
```

To set the limit for a group of users, list their names after the –u option. Separate multiple names with a comma or space (enclose the list in quotes if you use a space). *Priority–level* is a number from 0 to 39. The lower the number the higher the priority, or, in this case, the priority limit.

For All Other Users

If you want to set a priority limit for the remaining users, enter the following command:

```
# lpusers -q priority-level
```

This command sets the default limit; the default applies to users for whom you have not set a personal limit.

Revising Personal Priority Limits

If you later decide that someone should have a different priority limit, just re-enter the first command above with a new limit.

```
# lpusers -q priority-level -u user-name
```

If you decide that the default limit is more appropriate for someone who already has a personal limit, enter the following command:

```
# lpusers -u user-name
```

Again, you can do this for more than one user at a time by including a list of names.

Removing Personal Priority Limits

Using the **lpusers** command with just the **-u** option removes users' personal priority limits and puts the default limit into effect for those users.

Set a Default Priority

Do not confuse this default with the default limit. This default priority is used when a user does not specify a priority level; the default limit is applied if you have not assigned a limit for a user.

NOTE: If the default priority is greater than the limit for a user, the limit is used instead.

If you do not set a default priority, the LP Print Service uses a default of 20.

Procedure

To set the default priority (the priority level assigned to print requests submitted without a priority), use the following command:

```
# lpusers -d priority-level
```

Examine the Priority Limits and Defaults

You can examine all the settings you have assigned for priority limits and defaults by entering the following command:

```
# lpusers -l
```

Move a Request Around in the Queue

Once a user has submitted a print request, you can move it around in the queue to some degree:

- You can adjust the priority to any level, regardless of the limit for the user (who may adjust it only up to his or her limit)
- Both you and the user can put it on hold and allow other requests to be printed ahead of it
- You can put it at the head of the queue for immediate printing

Use the **lp**(1) command to do any of these tasks.

Changing the Priority for a Request

If you want to change the priority of a particular request that is still waiting to be printed, you can assign a new priority level to it. By doing so, you can move it in the queue so that it is ahead of lower priority requests, and behind requests at the same level or of higher priority. The priority limit assigned to the user (or the default priority limit) has no effect because, as the administrator, you can override this limit.

Enter the following command to change the priority of a request:

```
# lp -i request-ID -q new-priority-level
```

You can change only one request at a time with this command.

Putting a Request on Hold

Any request that has not finished printing can be put on hold. This action stops its printing, if it is currently printing, and keeps it from printing until you resume it. A user may also put his or her own request on hold and then resume it, but may not resume a print request that has been put on hold by the administrator.

To place a request on hold, enter the following command:

```
# lp -i request-ID -H hold
```

Enter the following command to resume the request:

```
# lp -i request-ID -H resume
```

Once resumed, a request continues to move up the queue and is eventually printed. If printing has already begun when you put it on hold, it is the next request printed.

Moving a Request to the Head of the Queue

You can move a print request to the head of the queue so that it is the next one eligible for printing. If you want it to start printing immediately but another request is currently being printed, you may interrupt the first request by putting it on hold, as described above.

Enter the following command to move a print request to the head of the queue:

```
# lp -i request-ID -H immediate
```

Only the administrator can move a request in this way.

NOTE: If you set more than one request for immediate printing, the requests are printed in the reverse order set; that is, the request moved to the head of the queue most recently is printed first.

Managing Printer Faults

NOTE: This section does not apply if you are making a remote printer accessible to users on your system.

The LP Print Service provides a framework for detecting printer faults and alerting you to them. Faults can range from simple problems, such as running out of paper or ribbon, or needing to replace the toner, to more serious faults, such as a local power failure or a printer failure. The range of fault indicators is also broad, ranging from dropping carrier (the signal that indicates that the printer is on line), to sending an XOFF, to sending a message.

Only two classes of printer fault indicators are recognized by the LP Print Service itself: a drop in carrier and an XOFF not followed in reasonable time by an XON. However, you can add filters that recognize any other printer fault indicators, and rely on the LP Print Service to alert you to a fault when the filter detects it.

NOTE: For a description of how to add a filter, see the "Providing Filters" section in this chapter. For a description of how a filter should let the LP Print Service know a fault has occurred, see the "Customizing the Print Service" section in the next chapter.

This section contains procedures for the following tasks:

- Receiving alerts
- Recovering from printer faults

Receive Alerts

You can choose one of several ways to receive an alert to a printer fault:

- By electronic mail. See the description of **mailx**(1) in the *Reference Manual.*
- By a message sent to any terminal on which you are logged in. See the description of **write**(1) in the *Reference Manual.*
- Through a program of your choice.
- You can receive no alerts.

NOTE: If you elect to receive no alerts, you need a way of finding out about the faults and fixing them; the LP Print Service does not continue to use a printer that has a fault.

You can also arrange to receive repeated alerts every few minutes until the fault is cleared. You can choose the rate of repeated alerts, or you can choose to receive only one alert per fault.

If you do not define an alert method, you receive mail once for each printer fault. If you define a method without the **-W** option, you are alerted once for each fault.

Without a filter that provides better fault detection, the LP Print Service can not automatically determine when a fault has been cleared except by trying to print another file. It assumes that a fault has been cleared when it is successfully able to print a file. Until that time, if you have asked for only one alert per fault, you do not receive another alert.

NOTE: If, after you have fixed a fault, but before the LP Print Service has tried printing another file, the printer faults again, or if your attempt to fix the fault fails, you are not notified. Receiving repeated alerts per fault, or requiring manual re–enabling of the printer overcomes this problem.

Receive Alerts

To receive alerts to a printer fault, enter one of the following commands:

```
# lpadmin -p printer-name -A mail -W minutes
# lpadmin -p printer-name -A write -W minutes
# lpadmin -p printer-name -A 'command' -W minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run the *command* for each alert. The *minutes* argument is the number of minutes between repeated alerts.

Receive No Alerts

If you do not want the LP Print Service to issue an alert when a fault occurs, enter the following:

```
# lpadmin -p printer-name -A none
```

NOTE: If you want mail sent or a message written to another user when a printer fault occurs, use the third command with the option **–A 'mail** *user–name*' or **–A 'write** *user–name*'.

If the *printer–name* is **all** in any of the commands above, the alerting condition applies to all printers.

Stop Alerts

Once a fault occurs and you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current fault only), by entering the following command:

```
# lpadmin -p printer-name -A quiet
```

Recover from Printer Faults

When a printer fault has been fixed and the printer is ready for printing again, the LP Print Service recovers in one of three ways:

- Printing at the top of the page where printing stopped
- Restarting printing at the beginning of the print request that was active when the fault occurred
- Waiting for you to tell the LP Print Service to re–enable the printer

NOTE: To continue printing at the top of the page where printing stopped, you must have a filter that can wait for a printer fault to be cleared before resuming properly. The default filter used by the LP Print Service cannot do this. If a proper filter is not being used, you are notified in an alert that recovery cannot proceed as you want.

Procedure

To specify the way the LP Print Service should recover after a fault is cleared, enter one of the following commands:

```
# lpadmin -p printer-name -F continue
# lpadmin -p printer-name -F beginning
# lpadmin -p printer-name -F wait
```

These commands direct the LP Print Service, respectively, to continue at the top of the page, restart from the beginning, or wait for you to enter an **enable** command to re–enable the printer (see the “Enable a Printer” section in the “Using the LP Print Service” chapter for information on the **enable** command).

Additional Information

If you do not specify how the LP Print Service is to resume after a printer fault, it tries to continue at the top of the page where printing stopped, or, failing that, at the beginning of the print request.

If the recovery is continue, but the interface program does not stay running so that it can detect when the printer fault has been cleared, printing is attempted every few minutes until it succeeds. You can force the LP Print Service to retry immediately by issuing an **enable** command.

Providing Forms

A form is a sheet of paper, on which text or graphical displays have already been printed, that can be loaded into a local printer (that is, a printer on your system) for use in place of plain paper. Common examples of forms include company letterhead, special paper stock, invoices, blank checks, vouchers, receipts, and labels.

The LP Print Service helps you manage the use of preprinted forms by keeping track of which print requests need special forms mounted and which forms are currently mounted. It can alert you to mount a new form.

This section tells you how you can manage the use of preprinted forms with the LP Print Service by:

- Adding a new form
- Changing the print service's description of an existing form
- Removing the print service's description of a form
- Examining the print service's description of a form
- Restricting user access to a form
- Alerting to the need to mount a form
- Informing the print service that a form has been mounted

Characteristics of a Form

When you add a form, you can define the following characteristics:

Page length
The length of the form, or of each page in a multi–page form. This can be expressed as the number of lines, or the size in inches or centimeters. Default is 66 lines.

Page width
The width of the form, expressed in characters, inches, or centimeters. Default is 80 characters.

Number of pages
The number of pages in a multi–page form. The LP Print Service uses this number with a filter (if available) to restrict the alignment pattern to the length of one form. (See the description of alignment patterns.) If no filter is available, the LP Print Service does not truncate the output. Default is 1 page.

Line pitch
A measurement of how closely together separate lines appear on the form. It can be expressed in either lines per inch or lines per centimeter. Default is 6 lines per inch.

Character pitch
A measurement of how closely together separate characters appear on the form. It can be expressed in either characters per inch or characters per centimeter. Default is 10 characters per inch.

Character set choice
The character set, print wheel, or font cartridge that should be used to print this form. A user may choose a different character set for his or her own print request when using this form, or you can insist that only one character set be used. Default is any.

Ribbon color

If the form should always be printed using a certain color ribbon, then the LP Print Service can remind you which color to use when you mount the form. Default is any.

Comment

Any comment you wish to make about the form. This comment is available for users to see so they can understand what the form is, when it should be used, and so on.

Alignment pattern

A sample file that the LP Print Service uses to fill one blank form. When mounting the form, you can print this pattern on the form to align it properly. You can also define a content type for this pattern so that the printer service knows how to print it.

NOTE: The LP Print Service does not try to mask sensitive information in an alignment pattern. If you do not want sensitive information printed on sample forms – when you align checks, for instance – then you should mask the appropriate data. The LP Print Service keeps the alignment pattern stored in a safe place, where only you (that is, the user **lp** and the superuser **root**) can read it.

Add a Form

Before you can use a form, you must let the system know that it exists and define its characteristics.

Procedure

Step 1: Gather information about the characteristics of your new form.

Step 2: You may want to record this information first in a separate file so you can edit it before entering it with **lpforms**. You can then use the file as input instead of typing each piece of information separately after a prompt.

Enter the information in the following format:

```
Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice:
character-set-name[,mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
alignment-pattern
```

The phrase **[mandatory]** is optional and, if present, means that the user can not override the character set choice in the form.

Content-type can be given optionally, with an alignment pattern. If you give this attribute, the print service uses the alignment pattern specified to determine how to filter and print the file.

With two exceptions, the listed information may appear in any order. The alignment pattern must always appear last, and the comment must always follow the line with the **Comment:** prompt. If the *comment* contains a line beginning with a key phrase (such as Page length, Page width, and so on), precede that line with a > character so the key phrase is hidden. Be aware, though, that any initial > is stripped from the comment when it is displayed.

Step 3: When you have gathered information about the form, enter it as input to the **lpforms** command. Use one of the following commands:

```
# lpforms -f form-name -F file-name
# lpforms -f form-name -
```

file–name is the full path for the file.

The first command gets the form definition from a file; the second command gets the form definition from you, through the standard input. A *form–name* can be anything you choose, as long as it contains a maximum of 14 alphanumeric characters and underscores.

Change a Form

If you need to change a form, just re-enter one of the above commands. You need only provide information for items that must be changed; items for which you do not specify new information stay the same.

```
# lpforms -f form-name -F file-name
# lpforms -f form-name -
```

Remove a Form

The LP Print Service imposes no fixed limit on the number of forms you may define. It is a good idea, however, to remove forms that are no longer appropriate.

To remove a form, enter the following command:

```
# lpforms -f form-name -x
```

Restrict User Access

If your system has a form that you do not want to make available to everyone, you can limit its availability to selected users. For example, you may want to limit access to checks to the people in the payroll department or accounts payable department.

Rules for Allow and Deny Lists

The LP Print Service restricts the availability of a form by using the list of users allowed or denied access to that form. If a user is not allowed to use a particular form, the LP Print Service rejects his or her request to print a file with it.

The method used to allow or deny users access to a form is similar to the method used to allow or deny users access to the **cron** and **at** facilities. (See the description of the **crontab**(1M) command in the *Reference Manual.*) Briefly, the rules are as follows:

- An allow list is a list of users who are allowed to use the form. A deny list is a list of users who are not allowed to use the form.
- If the allow list is not empty, only the users listed are allowed; the deny list is ignored. If the allow list is empty, the users listed in the deny list are not allowed to to use the form. If both lists are empty, there are no restrictions on who may use the form.
- Specifying **all** in the allow list allows everybody to use the form; specifying **all** in the deny list allows no one except the user **lp** and the superuser (**root**) to use the form.

Procedure

Add names of users to either list using one of the following commands:

```
# lpforms -f form-name -u allow:user-list
# lpforms -f form-name -u deny:user-list
```

The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list. Specifying **allow:all** allows everybody; specifying **deny:all** denies everybody.

The *user–list* is a comma or space separated list of names of users. If you use spaces to separate the names, enclose the entire list (including the allow: or deny: but not the **–u**) in quotes. Each item in the list can include a system name.

If you do not add user names to the allow or deny lists, the LP Print Service assumes that everybody may use the form.

Restricting Access to Forms on Remote Printers

If users on your system are to be able to access forms on a remote printer, all the users included on the allow list for the local system must be included on the allow list for the remote system, as well.

If, on the other hand, a local user is to be denied permission to use forms on a remote printer, the user's name does not need to appear in the deny lists on both the local and remote print services. As a courtesy to your users, however, make sure that any local users who are included in a deny list on a remote system are included in the corresponding deny list on your local system. Then whenever users on your system request a form that they are not authorized to use, they are immediately informed that permission to use the form is being denied.

Examine a Form

Once you define a form to the LP Print Service, you can examine it with one of two commands, depending on the type of information you want to check.

- **lpforms** displays the attributes of the form.
- **lpstat** displays the current status of the form.

Display the Attributes of a Form

Enter one of the following commands to display the attributes of a defined form:

```
# lpforms -f form-name -l
# lpforms -f form-name -l > file-name
```

These commands display the definition of the form; the second command captures this definition in a file, which you can use later to redefine the form if you inadvertently remove the form from the LP Print Service.

Display the Status of a Form

Enter one of the following commands to display the status of a defined form:

```
# lpstat -f form-name
# lpstat -f form-name -l
```

These commands display the status of the form, with the second of the two giving a long form of output, similar to the output of **lpforms –l**:

```
Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice: character-set[,mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

To protect potentially sensitive content, the alignment pattern is not shown if the **lpstat** command is used.

Mounting and Unmounting Forms

Before the LP Print Service can start printing files that need a preprinted form, you must physically mount the form on a printer, and notify the LP Print Service that you have mounted it. (It is not necessary for a form to be included on the allow list in order to mount it.) If alerting has been set on the form, you are alerted when enough print requests are queued waiting for it to be mounted. (See "Alert to Mount a Form".)

When you mount a form, you may want to see if it is lined up properly. If an alignment pattern is defined for the form, you can ask that it be repeatedly printed after you have mounted the form, until you have adjusted the printer so that the alignment is correct.

This section provides instructions for the following tasks:

- Mount forms
- Unmount forms
- Alert to mount a form

Mounting a form involves first loading it onto the printer and then telling the LP Print Service that the form is mounted. Because it is difficult to do this on a printer that is currently printing, and because the LP Print Service continues to print files not needing the form on the printer, you may have to disable the printer first. Thus, the proper procedure is to follow these three steps:

Step 1: Disable the printer, using the **disable** command.

Step 2: Mount the new form.

Step 3: Re-enable the printer, using the **enable** command. (See the "Enable a Printer" section of the "Using the LP Print Service" chapter.)

Mount the Form

Before the LP Print Service can start printing files that need a preprinted form, you must physically mount the form on a printer, and notify the LP Print Service that you have mounted it.

Step 1: Physically load the new form into the printer.

Step 2: Enter the following command to tell the LP Print Service the form has been mounted.

(The following command line is split into multiple lines for readability.)

```
# lpadmin -p printer-name -M -f form-name \
-a -o filebreak
```

You can drop the **–a** and **–o filebreak** options if you do not want to bother with the alignment pattern.

The **–o filebreak** option tells the LP Print Service to add a formfeed after each copy of the alignment pattern. The actual control sequence used for the formfeed depends on the printer involved and is obtained from the terminfo database. If the alignment pattern already includes a formfeed, leave out the **–o filebreak** option.

If you are mounting a form with an alignment pattern defined for it, you are asked to press the ENTER key before each copy of the alignment pattern is printed.

After the pattern is printed, you can adjust the printer and press the ENTER key again.

Unmount the Form

If you want to unmount a form, use the following command:

```
# lpadmin -p printer-name -M -f none
```

Alert to Mount a Form

If you define more forms than printers, you obviously can not print files on all the forms simultaneously. Some print requests may be held in a queue until you mount the forms they need. You can receive an alert when the number of requests waiting for a form has exceeded a specified threshold so that you know to mount the form.

Choose an Alert Method

You can choose one of several ways to receive an alert.

- By electronic mail. (See the description of the **mailx**(1) command in the *Reference Manual.*)

- Written to any terminal on which you are logged in. (See the description of the **write**(1) command in the *Reference Manual.*)
- Through a program of your choice.
- You can receive no alerts.

NOTE: If you elect to receive no alerts, you are responsible for checking to see if any print requests have not printed because the proper form is not mounted.

You can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the form is mounted. You can choose the rate of repeated alerts, or choose to receive only one alert for each form.

Once the form has been mounted and unmounted again, alerts resume if too many requests are waiting. Alerts also start again if the number of requests waiting falls below the –Q threshold and then rises up to the –Q threshold again. This happens when waiting requests are canceled and when the type of alerting is changed.

Receiving Alerts

To receive alerts to mount a form, enter one of the following commands:

```
# lpforms -f form-name -A mail -Q requests -W minutes
# lpforms -f form-name -A write -Q requests -W minutes
# lpforms -f form-name -A 'command' -Q requests -W \
minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run *command* for each alert.

In each command line, *requests* is the number of requests that need to be waiting for the form to trigger the alert, and *minutes* is the number of minutes between repeated alerts.

NOTE: If you want mail sent or a message written to another user when a printer fault occurs, use the third command with the option -A **'mail *user-name*'** or -A **'write *user-name*'**.

Receiving No Alerts

If you do not want the print service to issue an alert when the form needs to be mounted, enter the following command:

```
# lpforms -f form-name -A none
```

If *form-name* is **all** in any of the commands above, the alerting condition applies to all forms for which an alert has not already been defined.

Stopping Alerts

When you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current case only) by issuing the following command:

```
# lpforms -f form-name -A quiet
```

Providing Character Sets or Print Wheels

NOTE: Although local users may use character sets or print wheels that have been mounted on a remote printer (by the administrator of the remote system on which that printer resides), you can not mount a character set or a print wheel on a remote printer.

Printers differ in the way they can print in different font styles. Some have changeable print wheels, some have changeable font cartridges, others have pre–programmed, selectable character sets.

When adding a printer, you may specify what print wheels, font cartridges, or character sets are available with the printer.

Only one of these is assumed to apply to each printer. From the point of view of the LP Print Service, however, print wheels and changeable font cartridges are the same because they require you to intervene and mount a new print wheel or font cartridge. Thus, for ease of discussion, only print wheels and character sets are mentioned here.

NOTE: If you are adding a remote printer and you want your users to be able to use character sets or print wheels that have been mounted by the administrator of the remote system, you must list those character sets and print wheels, just as you would list the character sets and print wheels on a local printer.

Naming Character Sets and Print Wheels

When you list the print wheels or character sets available, you assign names to them. These names are for your convenience and the convenience of the users. Because different printers may have similar print wheels or character sets, you should use common names for all printers. Users can then submit a file for printing and ask for a particular font style, without regard for which printer is used or whether a print wheel or selectable character set is used.

You can either:

- Add print wheels
- Add and map character set names

If you do not add the print wheels or character sets that can be used with a printer, then the LP Print Service assumes the following:

- A printer that takes print wheels has only a single, fixed print wheel, and users may not ask for a special print wheel when using the printer
- A printer that has selectable character sets can take any character–set–name (csN) or terminfo name known for the printer

Add Print Wheels

If the printer has mountable print wheels, you need only add their names.

Procedure

To specify a list of print wheel names, enter the following command:

```
# lpadmin -p printer-name -S print-wheel-list
```

The *print–wheel–list* is a comma or space separated list of names. If you use spaces to separate the names, enclose the entire list (but not the –S) in quotes.

Add and Map Character Set Names

If the printer has selectable character sets, you need to add their names and map each one into a name or number that uniquely identifies it in the *terminfo* database.

Procedure

Use the following command to determine the names of the character sets listed in the *terminfo* database:

```
# tput -T printer-type csnm 0
```

Printer–type is the name of the printer type in question. The name of character set 0 (the character set obtained by default after the printer is initialized) is printed. Repeat the command, using 1, 2, 3, and so on in place of the 0, to see the names of the other character sets. In general, the *terminfo* names should closely match the names used in the user documentation for the printer. However, because not all manufacturers use the same names, the *terminfo* names may differ from one printer type to the next.

To list character set names and map them into *terminfo* names or numbers, enter the following command:

```
# lpadmin -p printer-name -S character-set-list
```

The *character–set–list* is also a comma or space separated list; however, each item in the list looks like one of the following:

```
csN=character-set-name
character-set-name1=character-set-name2
```

The *N* in the first case is a number from 0 to 63 that identifies the number of the character set in the terminfo database. The *character–set–name1* in the second case identifies the character set by its terminfo name. In either case the name to the right of the equal sign (=) is the name you may use as an alias of the character set.

NOTE: You do not have to provide a list of aliases for the character sets if the terminfo names are adequate. You may refer to a character set by number, by terminfo name, or by your alias.

Example

Your printer has two selectable character sets (sets #1 and #2) in addition to the standard character set (set #0). The printer type is **5310**. To determine the names of the selectable character sets, enter the following commands:

```
$ tput -T 5310 csnm 1
english
$ tput -T 5310 csnm 2
finnish
```

The words *english* and *finnish*, the output of the commands, are the names of the selectable character sets. You feel that the name *finnish* is adequate for referring to character set #2, but better names are needed for the standard set (set #0) and set #1.

Enter the following command to define synonyms:

```
# lpadmin -p printer-name -S "cs0=american,\
english=british"
```

The following three commands then produce identical results:

```
# lp -S cs1 -d any ...

# lp -S english -d any ...

# lp -S british -d any ...
```

Mounting and un–mounting Print Wheels

Before the LP Print Service can start printing files that need a print wheel, you must physically mount the print wheel on a printer and notify the LP Print Service that you have mounted it. If alerting has been set on the print wheel, you are alerted when enough print requests are queued and waiting for it to be mounted. (See "Alert to Mount a Print Wheel".)

This section provides instructions for the following tasks:

- Mount print wheels
- Unmount print wheels
- Alert to mount a print wheel

Mounting a print wheel involves first loading it onto the printer and then telling the LP Print Service that it is mounted. Because it is difficult to do this on a printer that is currently printing, you may have to disable the printer first. Thus, the proper procedure is to follow these three steps:

Step 1: Disable the printer, using the **disable** command.

Step 2: Mount the new print wheel.

Step 3: Re-enable the printer, using the **enable** command. (The **disable** and **enable** commands are described in the "Enable a Printer" section of the "Using the LP Print Service" chapter.)

Mount the Print Wheel

Before the LP Print Service can start printing files that need a print wheel, you must physically mount the print wheel on a printer, and notify the LP Print Service that you have mounted it.

Step 1: Physically load the new print wheel into the printer.

Step 2: Enter the following command to tell the LP Print Service the print wheel has been mounted.

```
# lpadmin -p printer-name -M -S \
print-wheel-name
```

Unmount the Print Wheel

If you want to unmount a print wheel, use the following command:

```
# lpadmin -p printer-name -M -S none
```

Alert to Mount a Print Wheel

NOTE: This section does not apply if you are making a remote printer available to users on your system.

Until a print wheel is mounted (see "Mount the Print Wheel" in this section), the system does not print a request for a print wheel. You could periodically monitor the number of print requests pending for a particular print wheel, but the LP Print Service provides an easier way: you can ask to be alerted when the number of requests waiting for a print wheel has exceeded a specified threshold.

Choose a Method of Alert

You can choose one of several ways to receive an alert.

- By electronic mail. See the description of the **mailx**(1) command in the *Reference Manual.*
- By a message sent to a terminal on which you are logged in. See the description of the **write**(1) command in the *Reference Manual.*
- Through a program of your choice.
- You can receive no alerts.

NOTE: If you elect to receive no alerts, you are responsible for checking to see whether any print requests have not printed because the proper print wheel is not mounted.

You can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the print wheel is mounted. You can choose the rate of repeated alerts, or you can opt to receive only one alert for each print wheel.

Receiving Alerts

To arrange for alerting to the need to mount a print wheel, enter one of the following commands:

(The following command lines are split into multiple lines for readability.)

```
# lpadmin -S print-wheel-name -A mail -Q \
requests -W  minutes
```

```
# lpadmin -S print-wheel-name -A write -Q \
requests -W minutes
```

```
# lpadmin -S print-wheel-name -A 'command' -Q \
requests -W minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run the *command* for each alert.

Requests is the number of requests that need to be waiting for the print wheel before the alert is triggered, and *minutes* is the number of minutes between repeated alerts.

Receiving No Alerts

If you do not want the print service to issue an alert when a print wheel needs to be mounted, enter the following:

```
# lpadmin -S print-wheel-name -A none
```

NOTE: If you want mail sent or a message written to another user when a printer fault occurs, use the third command with the option –A '**mail** *user–name*' or –A '**write** *user–name*'.

If *print–wheel–name* is all in any of the commands above, the alerting condition applies to all print wheels for which an alert has already been defined.

If you do not define an alert method for a print wheel, you do not receive an alert to mount it. If you do define a method without the **–W** option, you are alerted once for each occasion.

Once the print wheel has been mounted and unmounted again, alerts start again if too many requests are waiting. Alerts also start again if the number of requests waiting falls below the **–Q** threshold and then rises up to the **–Q** threshold again, as when waiting requests are canceled, or if the type of alerting is changed.

Stopping alerts

When you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current case only), by entering the following command:

```
# lpadmin -S print-wheel-name -A quiet
```

Providing Filters

This section explains how you can manage the use of filters with the LP Print Service. Specifically, you can:

- Define a new filter
- Change a filter
- Remove a filter
- Examine a filter
- Restore factory defaults

The "Customizing the Print Service" section in the "Adjusting the LP Print Service" chapter describes how you can write a filter First, let's see what a filter is and how the LP Print Service can use one.

A Word of Caution

Adding, changing, or deleting filters can cause print requests still queued to be canceled. Requests that are no longer printable, because a filter has been removed or changed, are canceled (with notifications sent to the people who submitted them). Responses to new or changed print requests may also be delayed when filters are changed, due to the many characteristics that must be evaluated for each print request still queued. These delays can become noticeable if there is a large number of requests that need to be filtered.

Because of this possible impact, you may want to make changes to filters during periods when the LP Print Service is not being used much.

What Is a Filter?

A filter is a program that you can use for any of three purposes:

- To convert a user's file from one data format to another so that it can be printed properly on a given printer
- To handle the special modes of printing that users may request with the **-y** option to the **lp** command (such as two-sided printing, landscape printing, draft or letter quality printing)
- To detect printer faults and notify the LP Print Service of them, so that the print service can alert you

Not every filter can perform all three tasks. Given the printer-specific nature of these three roles, the LP Print Service has been designed so that these roles can be implemented separately. This separation allows you or a printer manufacturer (or another source) to provide filters without having to change the LP Print Service.

A default filter is provided with the LP Print Service to provide simple printer fault detection; it does not convert files or handle any of the special modes. It may, however, be adequate for your needs.

Will Any Program Make a Good Filter?

It is tempting to use a program such as **troff**, **nroff**, or a similar word-processing program as a filter. However, the **troff** and **nroff** programs have a feature that allows references to be made in a source file to other files, known as include files. The LP Print Service does not recognize include files; it does not queue any that are referenced by a source file when that file is in a queue to be printed. As a result, the **troff** or **nroff** program, unable to access the include files, may fail. Other programs may have similar features that limit their use as filters.

Here are a few guidelines for evaluating a program for use as a filter:

- Make sure that only programs capable of reading data from standard input and writing data to standard output are used as filters.
- Examine the kinds of files users submit for printing that require processing by the program. If they stand alone (that is, if they do not reference other files that the program needs), the program is probably okay.
- Check also to see if the program expects any files other than those submitted by a user for printing. If it does, those files must be in the directory of the person using the filter, and they must be readable by all users authorized to use the filter.
- If referenced files are permitted in the files submitted for printing, or if the program needs files other than those submitted by a user, then the program, unable to access the additional files, is likely to fail. We suggest you do not use the program under consideration as a filter; instead, have users run the program before submitting files for printing.

Referenced files that are always specified by full path names *may* be okay, but only if the filter is used for local print requests. When used on requests submitted from a remote machine for printing on your machine, the filter may still fail if the referenced files exist only on the remote machine.

Define a Filter

When adding a new filter, you must first define the characteristics of its use. Issue the **lpfilter** command with arguments that specify the values of the following filter characteristics:

- The name of the filter (that is, a command name)
- The types of input it accepts
- The types of output it produces
- The types of printers to which it can send jobs
- The names of specific printers to which it can send jobs
- The type of the filter (whether it's a **fast** filter or a **slow** filter)
- Options

Each of these characteristics is described in this section.

Command
: The full path of the filter program. If there are any fixed options that the program always needs, include them here.

Input types
: The list of file content types that the filter can process. The LP Print Service does not impose a limit on the number of input types that can be accepted by a filter, but most filters can take only one.

  You can use whatever names you like here, using a maximum of 14 alphanumeric characters and dashes (not underscores). Because the LP Print Service uses these names to match a filter with a file type, you should follow a consistent naming convention. For example, if more than one filter can accept the same input type, use the same name for that input type when you specify it for each filter. Advertise these names to your users so they know how to identify the type of a file when submitting that file for printing. Default is any.

Output types

The list of file types that the filter can produce as output. For each input type the filter produces a single output type; the output type may vary, however, from job to job. The names of the output types are also restricted to 14 alphanumeric characters and dashes. These names should either match the types of printers you have on your system, or match the input types handled by other filters. Default is any.

Printer types

The list of printer types into which the filter can convert files. For most filters this list is identical to the list of output types, but it can be different. Default is any.

Printers

Specific printers the filter can print on. In most cases a filter should be able to work with all printers that accept its output, so you can usually skip this part of the filter definition. Default is any.

Filter type

The LP Print Service recognizes fast filters and slow filters. Fast filters incur little overhead in preparing a file for printing and must have access to the printer when they run. A filter that is to detect printer faults has to be a fast filter. Slow filters incur a lot of overhead in preparing a file and do not require access to a printer. The LP Print Service runs slow filters in the background, without tying up a printer. This allows files that do not need slow filtering to move ahead; printers are not left idle while a slow filter works on a file if other files can be printed simultaneously. Default is slow.

Options

Specify how different types of information should be transformed into command line arguments to the filter command. This information may include specifications from a user (with the print request), the printer definition, and the specifications implemented by any filters used to process the request.

There are 13 sources of information, each of which is represented by a keyword. Each option is defined in a template, a statement in the following format: *keyword pattern=replacement.* This type of statement is interpreted by the **lpfilter** command to mean, "When the information referred to by keyword has the value matched by pattern, take the replacement string, expand any regular expressions it contains, and append the result to the command line."

The options specified in a filter definition may include none, all, or any subset of these thirteen keywords. In addition, a single keyword may be defined more than once, if multiple definitions are required for a complete filter definition. (See "Defining Options with Templates".)

When you've gathered enough information to define these characteristics of your filter, you are ready to run the **lpfilter** command, using your data as arguments. Because there are so many arguments, and because you may need to enter some of them more than once (with different values), we recommend you record this information first in a separate file and edit it, if necessary. You can then use the file as input to the **lpfilter** command and avoid typing each piece of information separately.

Whether you store the information in a file or enter it directly on the command line, use the following format:

```
Command: command-pathname [options]
Input types: input-type-list
Output types: output-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: fast or slow
Options: template-list
```

The information can appear in any order. Not all the information has to be included.

Defining Options With Templates

A template is a statement in a filter definition that defines an option to be passed to the filter command based on the value of one of the characteristics of the filter. A filter definition may include more than one template. You may enter multiple templates on a single line, separating them with commas, or you may enter them on separate lines, preceded by the **Options:** prefix.

The format of a template is as follows:

```
keyword pattern = replacement
```

The *keyword* identifies the type of option being registered for a particular characteristic of the filter.

Let's look at an example of how an option is defined for a particular filter. Suppose you want to have the print service scheduler assign print requests to filters on the basis of the following criteria:

- If the type of **OUTPUT** to be produced by the filter is **impress**, then pass the **–I** option to the filter
- If the type of **OUTPUT** to be produced by the filter is **postscript**, then pass the **–P** option to the filter

To specify these criteria, provide the following templates as options to the **lpfilter** command.

```
Options: OUTPUT impress=-I, OUTPUT postscript=-P
```

If the **Options:** line becomes too long, put each template on a separate line, as follows:

```
Options: OUTPUT impress=-I
Options: OUTPUT postscript=-P
```

In both templates, the *keyword* is defined as **OUTPUT**. In the first template, the value of pattern is **impress** and the value of the replacement is **–I**. In the second template, the value of pattern is **postscript** and the value of the replacement is **–P**.

Template Keywords

Figure 5–1 lists and describes the keywords available for defining **Options** in a filter definition:

Characteristic	Keyword	Possible *patterns*	Example
Content type (input)	INPUT	content–type	troff
Content type (output)	OUTPUT	content–type	postscript
Printer type	TERM	printer–type	att495
Printer name	PRINTER	printer–name	lp1
Character pitch	CPI	scaled–decimal	10
Line pitch	LPI	scaled–decimal	6
Page length	LENGTH	scaled–decimal	66
Page width	WIDTH	scaled–decimal	80
Pages to print	PAGES	page–list	1–5,13–20
Character set	CHARSET	character–set	finnish
Form name	FORM	form–name	invoice2
Number of copies	COPIES	integer	3
Special modes	MODES	mode	landscape

Figure 5–1: Filter Keywords

To find out which values to supply for each type of template (that is, for the pattern and replacement arguments for each keyword), see the source of information listed below.

- The values for the **INPUT** and **OUTPUT** templates come from the file type that needs to be converted by the filter and the output type that has to be produced by the filter, respectively. They are each a type registered with the filter.
- The value for the **TERM** template is the printer type.
- The value for the **PRINTER** template is the name of the printer that is used to print the final output.
- The values for the **CPI, LPI, LENGTH,** and **WIDTH** templates come from the user's request, the form being used, or the default values for the printer.
- The value for the **PAGES** template is a list of pages that should be printed. Typically, it is a comma separated list of page ranges, each of which consists of a dash separated pair of numbers or a single number (such as 1–5,6,8,10 for pages 1 through 5, 6, 8, and 10). However, whatever value was given in the **–P** option to a print request is passed unchanged.
- The value for the **CHARSET** template is the name of the character set to be used.
- The value for the **FORM** template is the name of the form requested by the **–f** option of the **lp** command.
- The value of the **COPIES** template is the number of copies that should be made of the file. If the filter uses this template, the LP Print Service reduces to 1 the number of copies of the filtered file it prints, since this "single copy" is really the multiple copies produced by the filter.

- The value of the **MODES** template comes from the **–y** option of the **lp** command (the command used to submit a print request). Because a user can specify several **–y** options, there may be several values for the **MODES** template. The values are applied in the left–to–right order given by the user.

The *replacement* part of a template shows how the value of a template should be given to the filter program. It is typically a literal option, sometimes with the place–holder * included to show where the value goes. The pattern and replacement can also use the regular expression syntax of **ed**(1) for more complex conversion of user input options into filter options. All of the regular expression syntax of **ed**(1) is supported, including the \(... \) and \n constructions, which can be used to extract portions of the pattern for copying into the replacement, and the &, which can be used to copy the entire pattern into the replacement.

If a comma or an equals sign (=) is included in a pattern or a replacement, escape its special meaning by preceding it with a backslash (\). Note that some regular expressions include commas that have to be escaped this way. A backslash in front of any of these characters is removed when the pattern or replacement is used.

The following examples show how this replacement works.

Example 1

You provide the following filter definition for a filter called **col**.

```
Input types:  N37, Nlp, simple
Output types: simple
Command:      /usr/bin/col
Options:      TERM 450 = -b, MODES expand = -x
Options:      INPUT simple = -p -f
```

NOTE: If you provide more than one definition (that is, more than one line) for any filter characteristic other than **OPTIONS**, only the last definition is used by the print service.

After you register this definition with the print service by entering it as input with the **lpfilter** command, users' print requests are handled as follows:

- If a user enters the command

  ```
  # lp -y expand report.dec10
  ```

  the filter command runs with the following arguments:

  ```
  /usr/bin/col -x -p -f
  ```

- If a user enters the command

  ```
  # lp -T N37 -y  expand report.dec10
  ```

  the filter command runs with the following arguments:

  ```
  /usr/bin/col -x
  ```

  Qualifier: The default printer is not of type 450.

- If a user enters the command

  ```
  # lp -y expand -T 450 report.dec10
  ```

  the filter command runs with the following arguments:

  ```
  /usr/bin/col -b -x
  ```

Example 2

The filter program is called **/usr/lib/lp/postscript/dpost**. It takes one input type, **troff,** produces an output type called **postscript**, and works with any printer of type **postscript**. You decide that your users need give just the abbreviations **port** and **land** when they ask for the paper orientation to be portrait mode and landscape mode, respectively. Because these options are not intrinsic to the LP Print Service, users must specify them using the **–y** option to the **lp** command.

The filter definition would look like this:

```
Input types: troff
Output types: postscript
Printer types: postscript
Filter type: slow
Command: /usr/lib/lp/postscript/dpost
Options: LENGTH * = -l *, CHARSET * = -s *
Options: MODES port = -o portrait, MODES land = -o
landscape
```

A user submitting a file of type **troff** for printing on a PostScript printer (type **postscript**), with requests for landscape orientation and the **gothic** character set, would enter the following command:

 # lp -T troff -S gothic -y land -d any

Then this filter would be invoked by the LP Print Service to convert the file as follows:

/usr/lib/lp/postscript/dpost -S gothic -o landscape

Example 3

You add the following option template to the previous example:

Options: MODES size\=\([0-9]*\)x\([0-9]*\) = -h\1 -w\2

This template is used to convert a **MODES** option of the form:

-y size=*heightxwidth*

into a pair of filter options:

-h*height* -w*width*

So if a user gives the following command:

 # lp -y size=24x80

the **dpost** command would include the following options:

-h24 -w80

Add a Filter

Once a filter definition is complete, enter one of the following commands to add the filter to the system.

```
# lpfilter -f filter-name -F file-name
# lpfilter -f filter-name -
```

The first command gets the filter definition from a file, and the second command gets the filter definition from the standard input. A *filter-name* can be any string you choose, with a maximum of 14 alphanumeric characters and underscores.

Change a Filter

If you need to change a filter, just re-enter one of the same commands. You need only provide information for those items that must be changed; items for which you do not specify new information stay the same.

Remove a Filter

The LP Print Service imposes no fixed limit on the number of filters you can define. It is a good idea, however, to remove filters no longer applicable, to avoid extra processing by the LP Print Service which must examine all filters to find one that works in a given situation.
To remove a filter, enter the following command:

```
# lpfilter -f filter-name -x
```

Examine a Filter

Once you add a filter definition to the LP Print Service, you can examine it by running the **lpfilter** command. The output of this command is the filter definition displayed in a format that makes it suitable as input. You may want to save this output in a file that you can use later to redefine the filter if you inadvertently remove the filter from the LP Print Service.

To examine a defined filter, enter one of the following commands:

```
# lpfilter -f filter-name -l
# lpfilter -f filter-name -l >file-name
```

The first command presents the definition of the filter on your screen; the second command captures this definition in a file for future reference.

Restore Factory Defaults

The software is shipped from the factory with default values set for all options. If, after changing these defaults, you want to reset them, use this task.
Enter the following command:

```
# lpfilter -f filter-name -i
```

You can restore an individual filter by giving its name in place of *filter–name*, or you can restore everything by giving the name **all**.

Cleaning Out the Request Log

There are three log files contained in the */var/lp/logs* directory. These files are *lpsched, lpNet,* and *requests*. The files contain log information concerning the status of the LP spooler and the print jobs.

The three files are not removed by the LP Print Service, but can be cleaned out periodically, using, for instance, the **cron** facility. (See the description of the **crontab**(1M) command in the *Reference Manual.*)

An example of a **crontab** entry is shown below.

```
13 3 * * * cd /var/lp/logs; if [ -f requests ]; then
/usr/bin/mv requests xyzzy; /usr/bin/cp xyzzy
requests; >xyzzy;  /usr/lbin/agefile -c2 requests;
/usr/bin/mv xyzzy requests; fi
```

(This is one line in the **crontab** but is split into several lines here for readability.) What this entry does, briefly, is "age" the file, changing the name to requests–1, and moving the previous day's copy to requests–2. The number 2 in the **–c** option to the agefile program keeps the log files from the previous two days, discarding older log files. By changing this number you can change the amount of information saved. On the other hand, if you want the information to be saved more often, or if you want the file to be cleaned out more often than once a day, you can change the time when the crontab entry is run by changing the first two numbers. The current values, 13 and 3, cause cleaning up to be done at 3:13 A.M. each day.

The **crontab** entries for cleaning up the scheduler log files invokes the following shell scripts:

- */usr/lib/lp/bin/lp.daily* — cleans up files daily
- */usr/lib/lp/bin/lp.weekly* — cleans up files weekly

Note that the you can modify these shell scripts to suite your requriements. However, you should not use the **mv**(1) command to save the log files as some other files, since LP continues to log information into the moved file since the log files are not closed each time information is logged.

The request log has a simple structure that makes it easy to extract data from it using common UNIX shell commands. Requests are listed in the order they are printed, and are separated by lines showing their request IDs. Each line below the separator line is marked with a single letter that identifies the kind of information contained in that line. Each letter is separated from the data by a single space. See Figure 5–2, Figure 5–3, and Figure 5–4 for details.

Letter	Content of line
=	Separator line. Contains the request ID, the user and group IDs of the user, the total number of bytes in the original (unfiltered) files, and the time when the request was queued. These items are separated by commas and are in the order named here.
C	Number of copies printed.
D	Printer or class destination or the word **any**.
F	Name of the file printed. This line is repeated for each file printed. Files were printed in order given.
f	Name of the form used.

Figure 5–2: Information in Request Log (1 of 3)

Letter	Content of line
H	One of the three types of special handling: **resume**, **hold**, and **immediate**. The only useful value found in this line is **immediate**.
N	Type of alert used when the print request successfully completed: **M** if user was notified by mail; **W** if user was notified by a message to his or her terminal.
p	List of pages printed.
r	Single letter line included if user asked for "raw" processing of the files (the **–r** option of the **lp** command).
S	Character set or print wheel used.
s	Outcome of the request, shown as a combination of individual bits expressed in hexadecimal form. While several bits are used internally by the print service, the most important bits are listed below: 0x0004 Slow filtering finished successfully. 0x0010 Printing finished successfully. 0x0040 Request was canceled. 0x0100 Request failed filtering or printing.
T	Title placed on banner page.
t	Type of content found in file(s).
U	Name of user who submitted the print request.
x	Slow filter used for the request.
Y	List of special modes to give to filters used to print the request.

Figure 5–3: Information in Request Log (2 of 3)

Letter	Content of line
y	Fast filter used for the request.
z	Printer used for the request. This differs from the destination (the D line) if the request was queued for any printer or a class of printers, or if the request was moved to another destination by the LP Print Service administrator.

Figure 5–4: Information in Request Log (3 of 3)

Chapter 6

Adjusting the LP Print Service

This chapter provides information on adjusting the LP Print Service to meet other needs. It describes the following:

PostScript Printers

PostScript is a general purpose programming language, like C or Pascal. In addition to providing the usual features of a language, however, PostScript allows a programmer to specify the appearance of both text and graphics on a page.

A PostScript printer is equipped with a computer that runs an interpreter for processing PostScript language files. When a PostScript printer receives a file, it runs that file through the interpreter and then prints it. Unless special provisions have been made by the manufacturer, files submitted to a PostScript printer must be written in the PostScript language.

This section describes the following aspects of PostScript printers:

- PostScript facilities
- How to use a PostScript printer
- Supporting non–PostScript print requests
- Additional PostScript capabilities provided by filters
- The administrator's duties

PostScript Facilities

PostScript provides excellent facilities for managing and combining text and graphics. Specifically, it allows users to:

- Construct, position, and scale geometric figures with any orientation
- Specify a number of different fonts that can be placed on a page in any position, size, or orientation
- Readily combine text and graphics
- Proof draft copies on a low–resolution device and print the final version in higher resolution on a different device

Applications that support PostScript, including word–processing and publishing software, can create documents in the PostScript language without intervention by the user. Thus, users do not need to know the details of the language to take advantage of its features.

How to Use a PostScript Printer

When the PostScript printers and filters have been installed, LP manages PostScript files like any others. If *psfile* is a file containing a PostScript document and *psprinter* has been defined to LP as a PostScript printer, the following command schedules the print request and transmits the request to the PostScript printer:

```
# lp -d psprinter -Tpostscript psfile
```

Supporting Non–PostScript Print Requests Using Filters

Because PostScript is a language and PostScript printers are expecting print requests written in that language, some applications may produce standard print requests that may not be intelligible to PostScript printers. Figure 6–1 shows some examples of print requests that may not be interpreted by some PostScript printers.

Content Type	Type of Print Request
troff	Print a **troff** text file.
simple	Print an ASCII ("simple") text file.
dmd	Print the contents of a bit–mapped display from a terminal such as an AT&T 630.
tek4014	Print files formatted for a Tektronix 4014 device.
daisy	Print files intended for a Diablo 630 ("daisy–wheel") printer.
plot	Print plot–formatted files

Figure 6–1: Non-PostScript Print Requests

Optional filters provided with the LP Print Service translate print requests with these formats to the PostScript language. For example, to convert a file containing ASCII or **troff** code to PostScript code, the filter takes that text and writes a program around it, specifying printing parameters such as fonts and the text layout on a page.

Once the PostScript filters are installed, they are invoked automatically by the LP Print Service when a user specifies a content–type for a print request with the **-T** option. For example,

```
# lp -d psprinter -T simple report2
```

automatically converts the ASCII file *report2* (a file with an ASCII or simple format) to PostScript when the destination printer *psprinter* has been defined to the system as a PostScript printer.

Additional PostScript Capabilities Provided by Filters

The filters previously described also take advantage of PostScript capabilities to provide additional printing flexibility. You may access most of these features through the mode option (invoked by the –y option) to the **lp** command. These filters allow you to use the following unusual options for your print jobs:

–y landscape
Change the orientation of a physical page from portrait to landscape.

–y x=x*number*, y=y*number*
Change the default position of a logical page on a physical page by moving the origin.

–y group=*number*
Group multiple logical pages on a single physical page.

–P *number*
Select, by page numbers, a subset of a document to be printed.

–T matrix
Print a gray–scale representation of a matrix. (A gray–scale matrix is a matrix in which each cell is colored one of seven shades of gray to indicate the value of the cell. Darker shades correspond to larger values.)

Reversed Pages
To print pages in reverse order, configure the PostScript printer type as **PSR**, which invokes the postreverse procedure to reverse the pages.

NOTE: If filters are to be used with an application that creates PostScript output, make sure that the application conforms to the PostScript file structuring comments. In particular, the beginning of each PostScript page must be marked by the following comment :

```
%%Page:label ordinal
```

Ordinal is a positive integer that specifies the position of the page in the sequence of pages in the document.

For example, you have a file called *report2* that has a content type simple (meaning that the content of this file is in ASCII format). You want to print six pages of this file (pages 4–9) in landscape mode (that is, sideways on the page) with two logical pages on each physical page. Because one of the printers on your system (psprinter) is a PostScript printer, you can do this by entering the following command:

```
# lp -d psprinter -T simple -P 4-9 -y \
landscape,group=2 myfile
```

The Administrator's Duties

You must define the PostScript printers to the system with the **lpadmin** command and install the appropriate software to manage them. If you want to support fonts on your PostScript printers, you must perform additional tasks.

When you install the LP Print Service utilities (**lp** package), the system asks whether you will install PostScript printers. If you enter **yes**; the appropriate filters are installed.

Installing and Maintaining PostScript Printers

You install PostScript printers, like other printers, with the **lpadmin** command.

NOTE: The content type of a PostScript printer must be consistent with the content type used in PostScript filters. Therefore, you should install your PostScript printers with a content type of postscript and a printer type of PS. The content type accepted by the printer is specified with the –I option in the **lpadmin** command.

In addition, you must tell the LP Print Service which fonts are available on the printer. To do so, enter this information in the font list for each printer.

Installing and Maintaining PostScript Filters

PostScript filters are provided with UNIX System V Release 4 but installing them is optional. You can install them during regular installation by specifying PostScript printer support.

PostScript filters are contained in the */usr/lib/lp/postscript* directory.

Before any system and a PostScript printer can communicate, the **postprint** filter must be present on the system. This program is the only mandatory PostScript filter that communicates directly with the PostScript printer. The filters in Figure 6–2 allow other types of documents to be translated to PostScript and to be printed on a PostScript printer.

Filter Content Type	Filter
simple	postprint
troff	dpost
daisy	postdaisy
dmd(AT&T 630)	postdmd
tek4014	posttek
plot	postplot

Figure 6–2: PostScript Translation Filters

The filters in Figure 6–3 perform special functions:

Filter	Function
download	Download fonts
postreverse	Reverse or select pages
postmd	Matrix gray scales

Figure 6–3: Special PostScript Filters

Installing and Maintaining PostScript Fonts

One advantage of PostScript is its ability to manage fonts. Fonts are stored in outline form, either on the printer or on a computer that communicates with a printer. When a document is printed, the PostScript interpreter generates each character as needed (in the appropriate size) from the outline description of it. If a font required for a document is not stored on the printer being used, it must be transmitted to that printer before the document can be printed. This transmission process is called downloading fonts.

Fonts are stored and accessed in several ways.

- Fonts may be stored permanently on a printer. These printer resident fonts may be installed in ROM on the printer by the manufacturer. If the printer has a disk, fonts may be installed on that disk by you (that is, by the print service administrator). Most PostScript printers are shipped with thirty–five standard fonts.

- A font may be permanently down–loaded by being transmitted to a printer with a PostScript exitserver program. The font remains in the printer's memory until the printer is turned off. Use of exitserver programs requires the printer system password and may be reserved for the printer administrator. This method is useful when the majority of print requests serviced by that printer continually use a font.

- Fonts may be added to a user's print request by the user, and be transmitted as part of the user's print request. When the user's document has been printed, the space allocated to the font is freed for other print requests. The font is stored in the user's directory. This method is preferable for fonts with more limited usage.

- Fonts may be stored on a system shared by many users. Such fonts may be called host resident. This method is useful when a large number of fonts are available or when these fonts are not continually used by all print requests. If the fonts are used only on printers attached to a server, they should be stored on the server. If the fonts are used by users on one system who may send jobs to multiple printers on a network, they may be stored on the users' system.

Managing Printer–Resident Fonts

Most PostScript printers come equipped with fonts resident in the printer ROM. Some printers have a disk on which additional fonts are stored. When a printer is installed, the list of printer–resident fonts should be added to the font–list for that printer. These lists are kept in the printer administration directories. For a particular printer, this list is contained in the file */etc/lp/printers/printer–name/residentfonts* where *printer–name* is the name of the printer.

When fonts are permanently down–loaded to the printer, the font names should be added to this file. If the printer is attached to a remote system, this list should include fonts which reside on that system and are available for downloading to the printer. This prevents fonts from being transmitted unnecessarily across a network. These files must be edited manually; that is, with the help of a text editor such as vi.

Installing and Maintaining Host–Resident Fonts

Some fonts are resident on the host and transmitted to the printer as needed for particular print requests. As the administrator, you must make PostScript fonts available to all the users on a system.

Because fonts are requested by name and stored in files, LP keeps a map file with the correspondence between font names and the names of the files that contain the fonts. Both of these must be updated when fonts are installed on the host.

Install host–resident PostScript fonts by doing the following:

- Copy the font file to the appropriate directory.
- Add to the map table the name of the font and the name of the file in which it resides.

Where Are Fonts Stored?

The fonts available for use with PostScript printers reside in the */usr/share/lib/lp/hostfontdir/postscript/typeface/font* directories, where *typeface* is replaced by a name such as **palatino** or **helvetica**, and font is replaced by a name such as **bold** or **italic**.

Adding an Entry to the Map Table

Also within the *postscript* directory, you (the administrator) must create and maintain a map table that shows the correspondence between the name assigned to each font by the foundry (the company that created the font) and the name of the file in which that font resides. For example, to map the font called "Palatino Bold," add the following line to the map table:

```
Palatino-Bold/usr/share/lib/lp/hostfontdir/
palatino/bold
```

Once this entry exists in the map table on your system, your users can use the Palatino Bold font for their print jobs.

When they submit for printing a file containing a request for this font, the LP Print Service adds a copy of the file *usr/share/lib/lp/hostfontdir/palatino/bold* to that file before sending it to the printer.

Downloading Host–Resident Fonts

The creators of the PostScript language anticipated that users would want to download fonts to printers and provided a standard set of structuring conventions. The download filter relies on these structuring conventions to determine which fonts must be down–loaded.

When the LP Print Service receives a request for a job that requires fonts not loaded on the printer, it forwards that request to a filter that down–loads fonts. The request to download is made through the **–y download** option. Alternatively, an administrator may reinstall this filter so that all PostScript documents are examined to determine if fonts are needed. The –y option is then unnecessary.

The download filter does five things:

- It searches the PostScript document to determine which fonts have been requested. These requests are documented with the following PostScript structuring comments in the header comments:

  ```
  %%DocumentFonts: font1 font2 ...
  ```

- It searches the list of fonts resident on that printer to see if the requested font must be downloaded.
- If the font is not resident on the printer, it searches the host–resident font directory (by getting the appropriate file name from the map table) to see if the requested font is available.
- If the font is available, the filter takes the file for that font and adds it to the file to be printed.
- The filter sends the font definition file and the source file (the file to be printed) to the PostScript printer

Customizing the Print Service

Although the LP Print Service has been designed to be flexible enough to handle most printers and printing needs, it does not handle every possible situation. You may buy a printer that does not quite fit into the way the LP Print Service handles printers, or you may have a printing need that the standard features of the LP Print Service do not accommodate.

You can customize the LP Print Service in a few ways. This section tells you how you can do the following:

- Adjust the printer port characteristics
- Adjust the terminfo database
- Write an interface program
- Write a filter

Adjust the Printer Port Characteristics

You should make sure that the printer port characteristics set by the LP Print Service match the printer communication settings. The standard printer port settings have been designed to work with typical files and many printers, but they do not work with all files and printers.

When you add a new printer, read the documentation that comes with it so that you understand what it expects from the host (the LP Print Service). Then read the manual page for the stty(1) command in the *Reference Manual*. It summarizes the various characteristics that can be set on a terminal or printer port.

Only some of the characteristics listed in the **stty(1)** manual page are important for printers. The ones likely to be of interest to you are listed in Figure 6–4 (but you should still consult the **stty(1)** manual page for others).

stty Option	Meaning
evenp	Send even parity in the 8th bit
oddp	Send odd parity in the 8th bit
–parity	Do not generate parity; send all 8 bits unchanged
110 – 38400	Set the communications speed to this baud rate
ixon	Enable XON/XOFF (also known as START/STOP or DC1/DC3) flow control
–ixon	Turn off XON/XOFF flow control
–opost	Do not do any "output post–processing"
opost	Do "output post–processing" according to the settings listed below
onlcr	Send a carriage return before every linefeed
–onclr	Do not send a carriage return before every linefeed
ocrnl	Change carriage returns into linefeeds
–ocrnl	Do not change carriage returns into linefeeds
–tabs	Change tabs into an equivalent number of spaces
tabs	Do not change tabs into spaces

Figure 6–4: Printer Port Characteristics

When you have a set of printer port characteristics you think should apply, adjust the printer configuration. You may find that the default settings are sufficient for your printer.

Adjust the terminfo Database

The LP Print Service relies on a standard interface and the terminfo database to initialize each printer and establish a selected page size, character pitch, line pitch, and character set. Thus, it is usually sufficient to have the correct entry in the terminfo database to add a new printer to the LP Print Service. Several entries for AT&T printers and other popular printers are delivered in the standard terminfo database.

Each printer is identified in the terminfo database with a short name; this kind of name is identical to the kind of name used to set the TERM shell variable. For instance, the AT&T model 455 printer is identified by the name 455.

If you cannot find a terminfo entry for your printer, you should add one. If you do not, you may still be able to use the printer with the LP Print Service but you do not have the option of automatic selection of page size, pitch, and character sets, and you may have trouble keeping the printer set in the correct modes for each print request. Another option is to customize the interface program used with the printer. (See the "Write an Interface Program" section for details on how to do this.)

Hundreds of items can be defined for each terminal or printer in the terminfo database. However, the LP Print Service uses fewer than 50 of these. Figure 6–5, Figure 6–6, and Figure 6–7 list the items that need to be defined (as appropriate for the printer) to add a new printer to the LP Print Service.

terminfo item	Meaning
Booleans:	
cpix	Changing character pitch changes resolution
daisy	Printer needs operator to change character set
lpix	Changing line pitch changes resolution
Numbers:	
bufsz	Number of bytes buffered before printing
cols	Number of columns in a line
cps	Average print rate in characters per second
it	Tabs initially every # spaces
lines	Number of lines on a page
orc	Horizontal resolution in units per character
orhi	Horizontal resolution in units per inch
orl	Vertical resolution in units per line
orvi	Vertical resolution in units per inch

Figure 6–5: Printer Definitions in *terminfo* (1 of 3)

terminfo item	Meaning
	Strings
chr	Change horizontal resolution
cpi	Change number of characters per inch
cr	Carriage return
csnm	List of character set names
cud1	Down one line
cud	Move carriage down # lines
cuf	Move carriage right # columns
cuf1	Carriage right
cvr	Change vertical resolution
ff	Page eject
hpa	Horizontal position absolute
ht	Tab to next 8–space tab stop
if	Name of initialization file
iprog	Pathname of initializing program
is1	Printer initialization string
is2	Printer initialization string
is3	Printer initialization string
lpi	Change number of lines per inch
mgc	Clear all margins (top, bottom, and sides)
rep	Repeat a character # times
rwidm	Disable double wide printing
scs	Select character set
scsd	Start definition of a character set
slines	Set page length to # lines

Figure 6–6: Printer Definitions in *terminfo* (2 of 3)

terminfo item	Meaning
smgb	Set bottom margin at current line
smgbp	Set bottom margin
smgl	Set left margin at current column
smglp	Set left margin
smgr	Set right margin at current column
smgrp	Set right margin
smgt	Set top margin at current line
smgtp	Set top margin
swidm	Enable double wide printing
vpa	Vertical position absolute

Figure 6–7: Printer Definitions in *terminfo* (3 of 3)

To construct a database entry for a new printer, see details about the structure of the terminfo database in the **terminfo**(4) manual page .

Once you have made the new entry, you need to compile it into the database using the **tic**(1) program. Just enter the following command:

```
# tic file-name
```

file–name is the name of the file containing the terminfo entry you have crafted for the new printer.

NOTE: The LP Print Service gains much efficiency by caching information from the terminfo database. If you add or delete terminfo entries, or change the values that govern pitch settings, page width and length, or character sets, you should stop and restart the LP Print Service so it can read the new information.

Write an Interface Program

What Is an Interface Program?

NOTE: This section does not apply if you are making a remote printer accessible to users on your system.

The LP Print Service uses an interface program to manage the printer each time a file is printed. It has several tasks:

- To initialize the printer port (the connection between the computer and the printer)
- To initialize the printer (restore it to a normal state in case a previously printed file has left it in an unusual state) and set the character pitch, line pitch, page size, and character set requested by the user
- To print a banner page
- To run a filter that prepares the file for printing
- To manage printer faults

An interface program is not responsible for opening the printer port. The LP Print Service opens the port, a process which includes calling a dial–up printer, if one is used to connect the printer. The printer port connection is given to the interface program as standard output, and the printer is identified as the controlling terminal for the interface program so that a hangup of the port causes a SIGHUP signal to be sent to the interface program.

A customized interface program must not terminate the connection to the printer or reset the printer in any way.

If you do not choose an interface program, the standard one provided with the LP Print Service is used. This should be sufficient for most of your printing needs. If you prefer, however, you can change it to suit your needs, or completely rewrite your own interface program, and then specify it when you add a new printer.

If you are using a different interface program on a local printer, you can refer to it either by specifying its full path name or by referring to another printer using the same interface program.

To identify a customized interface program by name, specify the printer name and the path name of the interface program, as follows:

```
# lpadmin -p printer-name -i path-name
```

To use a customized interface program of another printer, specify the printer names as follows:

```
# lpadmin -p printer-name1 -e printer-name2
```

Printer-name1 is the name of the printer you are adding; *printer-name2* is the name of an existing printer that is using the customized interface program.

How to Write an Interface Program

NOTE: If you have an interface program that you have used with the LP Spooling Utilities before UNIX System V Release 3.2, it should still work with the LP Print Service. Note, though, that several **–o** options have been standardized, and are passed to every interface program. These may interfere with similarly named options used by your interface.

If you have a printer that is not supported by simply adding an entry to the terminfo database, or if you have printing needs that are not supported by the standard interface program, you can furnish your own interface program. It is a good idea to start with the standard interface program, and change it to fit, rather than starting from scratch. You can find a copy of it under the name */etc/lp/model*.

When the LP Print Service routes an output request to a printer, the interface program for the printer is invoked as follows:

```
#/var/spool/lp/admins/lp/interfaces/P ID user title
copies options file1 file2 ...
```

Arguments for the interface program are as follows:

P
Printer name

id
Request ID returned by the **lp**(1) command

user
Login name of the user who made the request

title
Optional title specified by the user

copies
Number of copies requested by the user

options
Blank-separated list of options specified by the user or set by the LP Print Service

file1,file2, ...
Full path names of the files to be printed

When the interface program is invoked, its standard input comes from */dev/null,* its standard output is directed to the printer port, and its standard error output is directed to a file that is given to the user who submitted the print request.

The standard interface recognizes the following values in the blank–separated list in *options*.

nobanner
Skips the printing of a banner page; without it, a banner page is printed.

nofilebreak
Skips page breaks between separate data files; without it, a page break is made between each file in the content of a print request.

cpi=*decimal-number1*
Specify a format of *decimal-number1* columns per inch. The words **pica, elite,** and **compressed** are acceptable replacements for *decimal-number1* and are synonyms, respectively, for 10 columns per inch, 12 columns per inch, and as many columns per inch as possible.

lpi=*decimal-number2*
Specify a format of *decimal-number1* columns per inch and *decimal-number2* lines per inch, respectively.

length=*decimal-number1*
Specify the length of the pages to be printed.

width=*decimal-number2*
Specify the width of the pages to be printed.

stty=*'stty-option-list'*
The *stty-option-list* is applied after a default *stty-option-list* as a set of arguments to the **stty** command. The default list is used to establish a default port configuration; the additional list given to the interface program is used to change the configuration as needed.

lpd=*'argument-list'*
Used internally by the **lpsched** command; you can ignore it.

flist=*'file-list'*
Used internally by the **lpsched** command; you can ignore it.

You may specify the above options when issuing a print request. Alternatively, the administrator may specify them to the LP Print Service as defaults either for the printer (**cpi, lpi, length, width, stty**) or for the preprinted form used in the request (**cpi, lpi, length, width**).

The following shell variables pass additional printer configuration information to the interface program:

TERM=*printer–type*
: Specifies the type of printer. The value is used as a key for getting printer capability information from the *terminfo* database.

FILTER=*'pipeline'*
: Specifies the filter to use to send the request content to the printer; the filter is given control of the printer.

CHARSET=*character–set*
: Specifies the character set to be used when printing the content of a print request.

A customized interface program should either ignore these options and shell variables or should recognize them and treat them in a consistent manner.

Customizing the Interface Program

Make sure that the custom interface program sets the proper stty modes (terminal characteristics such as baud rate and output options). The standard interface program does this, and you can follow suit. Look for the section that begins with the following shell comment:

```
## Initialize the printer port
```

Follow the code used in the standard interface program. It sets both the default modes and the adjusted modes given by either the LP Print Service or the user with a line such as the following:

```
stty mode options 0<&1
```

This command line takes the standard input for the **stty** command from the printer port. An example of an **stty** command line that sets the baud rate at 1200 and sets some of the option modes is as follows:

```
# stty -parenb -parodd 1200 cs8 cread clocal \
ixon 0<&1
```

One printer port characteristic not set by the standard interface program is hardware flow control. The way that this characteristic is set varies, depending on your computer hardware. The code for the standard interface program suggests where this and other printer port characteristics can be set. Look for the section that begins with the following shell comment:

```
# Here you may want to add other port \
initialization code.
```

Because different printers have different numbers of columns, make sure the header and trailer for your interface program correspond to your printer. Look in the code for the standard interface program for the section that begins with the following shell comment:

```
# Print the banner page
```

The custom interface program should print all user related error messages on the standard output or on the standard error. The messages sent to the standard error are mailed to the user; the messages printed on the standard output end up on the printed page where they can be read by the user when he or she picks up the output. Note that messages sent to standard error will be mailed to the user only when the interface program exits with a valid exit code.

When printing is complete, your interface program should exit with a code that shows that status of the print job. Exit codes are interpreted by the LP Print Service as shown in Figure 6–8:

Code	Meaning to the LP Print Service
0	The print request has been completed successfully.
1 to 127	A problem has been encountered in printing this particular request (for example, too many non–printable characters, or the request exceeds the printer capabilities). The LP Print Service notifies users by **mail** that there was an error in printing it. This problem does not affect future print requests.
128	Reserved for internal use by the LP Print Service. Interface programs must not exit with this code.
129	A printer fault has been encountered in printing the request. This problem affects future print requests. If the fault recovery for the printer directs the LP Print Service to wait for the administrator to fix the problem, the LP Print Service disables the printer. If the fault recovery is to continue printing, the LP Print Service does not disable the printer, but tries printing again in a few minutes.
greater than 129	These codes are reserved for internal use by the LP Print Service. Interface programs must not exit with codes in this range.

Figure 6–8: Exit Codes for Interface Programs

As the table shows, the interface program can alert the administrator to a printer fault by exiting with a code of 129. Unfortunately, if the interface program exits, the LP Print Service reprints the request from the beginning when the fault has been cleared. The interface program can also send a fault message to the LP Print Service but wait for the fault to clear When the fault clears, the interface program can resume printing the user's file. When the printing is finished, the interface program can give a zero exit code just as if the fault had never occurred. The interface program can also detect when the fault is cleared automatically, so that the administrator does not have to enable the printer.

The **lp.tell** program can also send fault messages. This is referenced using the $LPTELL shell variable in the standard interface code. The program takes its standard input and sends it to the LP Print Service where it is put into the message that alerts the administrator to the printer fault. If its standard input is empty, **lp.tell** does not initiate an alert. Examine the standard interface code immediately after these comments for an example of how the **lp.tell** ($LPTELL) program is used:

```
# Here's where we set up the $LPTELL program
# to capture fault messages.
# Here's where we print the file.
```

If the LP Print Service has to interrupt the printing of a file at any time, it "kills" the interface program with a signal TERM (trap number 15; see **kill**(1) and **signal**(2) in the *Reference Manual*). If the interface program dies from receiving any other signal, the LP Print Service assumes that future print requests are not affected, and continues to use the printer. The LP Print Service notifies the user that the request has not been finished successfully.

When the interface is first invoked, the signals **HUP, INT, QUIT,** and **PIPE** (trap numbers 1, 2, 3, and 13) are ignored. The standard interface traps these signals at appropriate times. The standard interface interprets these signals as warnings that the printer has a problem; when it receives one, it issues a fault alert.

Write a Filter

A filter is used by the LP Print Service each time it has to print a type of file that is not acceptable by a printer. A filter can be as simple or as complex as needed; there are only a few external requirements:

- The filter should get the content of a user's file from its standard input and send the converted file to the standard output.
- A **slow** filter can send messages about errors in the file to standard error. A **fast** filter should not, as described below. Error messages from a **slow** filter are collected and sent to the user who submitted the file for printing.
- If a **slow** filter dies because of receiving a signal, the print request is stopped and the user who submitted the request is notified. Likewise, if a **slow** filter exits with a non–zero exit code, the print request is stopped and the user is notified. The exit codes from **fast** filters are treated differently, as described below.
- A filter should not depend on other files that normally would not be accessible to a regular user; if a filter fails when run directly by a user, it will fail when run by the LP Print Service.

The "Providing Filters" section in the "Managing the LP Print Service" chapter describes how to add a filter to the LP Print Service.

If you want your filter to detect printer faults, you must also fulfill the following requirements:

- If possible, the filter should wait for a fault to be cleared before exiting. Additionally, it should continue printing at the top of the page where printing stopped after the fault clears. If the administrator does not want this contingency followed, the LP Print Service can stop the filter before alerting the administrator.

- It should send printer fault messages to its standard error as soon as the fault is recognized. It does not have to exit, but can wait as described above.

- It should *not* send messages about errors in the file to standard error. These should be included in the standard output stream, where they can be read by the user.

- It should exit with a zero exit code if the user's file is finished (even if errors in the file have prevented it from being printed correctly).

- It should exit with a non–zero exit code *only* if a printer fault has prevented it from finishing a file or an error message needs to be mailed to the user.

- When added to the filter table, it must be added as a fast filter. (See the "Providing Filters" section in the "Managing the LP Print Service" chapter for details.)

Chapter 7

Network Selection

This chapter describes the administrative command–level interface for Network Selection. It contains information about the following:

Network Selection

Network Selection allows UNIX Systems to communicate both with other UNIX systems and with non–UNIX systems. It provides a consistent way to determine what transport providers are installed on a machine and a standardized mechanism for finding service addresses.

The UNIX System V Release 4 Network Selection component is built around:

- A network configuration database (the*/etc/netconfig* file) that contains entries for each network available to the system
- An optional **NETPATH** environment variable, set by a user or the system administrator and containing an ordered list of network identifiers. These network identifiers match the *netconfig networkID* field and are used as links to the records in the *netconfig* file.

Users and administrators can edit these two components to customize the default list of networks an application tries to connect to.

As system administrator, you are responsible for:

- Maintaining the network configuration database file (*/etc/netconfig*) used by these routines
- Creating and maintaining the host and service files required for each of the Name–to–Address Mapping libraries

Caution

When editing sensitive system files, be sure to keep a backup copy of the file you are editing. When you have finished editing the file, use **diff**(1) on the edited file and the backup copy to verify that only the changes you want have been made.

The Network Selection application programming interface consists of a set of network configuration database access routines. One group of these library routines accesses only the *netconfig* entries identified by the **NETPATH** environment variable; another group of routines accesses *netconfig* directly.

The netconfig File

The *netconfig* file is a network configuration database that contains entries for each network available to the system.

You are responsible for maintaining the network configuration database file */etc/netconfig*. Entries in the netconfig file contain the following fields, in the order shown in Figure 7–1:

network ID	*semantics*	*flag*	*protocol family*	*protocol name*	*network device*	*directory lookup libraries*

Figure 7–1: Fields in *netconfig* Entries

The fields correspond to elements of the **struct netconfig** structure. Pointers returned by Network Selection library routines are pointers to *netconfig* entries in *struct netconfig* format. The *netconfig* file is described in the manual page **netconfig(4)**.

All symbolic names, structure definitions, and constant values for the Network Selection feature are defined in the header file */usr/include/sys/netconfig.h*.

netconfig fields

network ID

Identifies a network. The network ID:

- consists of non–NULL characters
- has a length of at least 1
- has no maximum length specified

This namespace is locally–significant and the local system administrator is the naming authority responsible for ensuring that all network IDs on a system are unique.

semantics

Identifies the semantics of the network, that is, the set of services it supports, by identifying the service interface it provides.

The semantics field is mandatory. The following semantics are recognized:

- **tpi_clts**—Transport Provider Interface, connectionless
- **tpi_cots**—Transport Provider Interface, connection–oriented
- **tpi_cots_ord**—Transport Provider Interface, connection–oriented and supports orderly release

flag

Records certain two–valued (true and false) attributes of networks. *flag* is a string composed of a combination of characters, each of which specifies the value of the corresponding attribute. If the character is present, the attribute is true. If the character is absent, the attribute is false. A hyphen (–) specifies that none of the attributes is present. Only one character is currently recognized:

- v — Visible (default) network. Used when the environment variable **NETPATH** is unset.

protocol family

Identifies a specific protocol family. The protocol family identifier follows the rules for *network IDs*:

- Consists of non–NULL characters
- Has a length of at least 1
- Has no maximum length specified

A hyphen (–) in the protocol family field indicates that none of the available protocol family identifiers applies, that is, the network is experimental. An application that wants to have family characteristics can match on the protocol family field when selecting a network (for example, an application can search for an osi family). In this case, the application is not protocol independent, since it has searched only for OSI entries.

The following are examples of protocol family identifiers:

- loopback — Loopback (local to host)
- inet — Internetwork: UDP, TCP, etc.
- implink — ARPANET imp addresses
- pup — PUP protocols; for example, BSP
- chaos — MIT CHAOS protocols
- ns — XEROX NS protocols
- nbs — NBS protocols
- ecma — European Computer Manufacturers Association
- datakit — DATAKIT protocols

- ccitt — CCITT protocols, X.25, etc.
- sna — IBM SNA
- decnet — DECNET
- dli — Direct data link interface
- lat — LAT
- hylink — NSC Hyperchannel
- appletalk — Apple Talk
- nit — Network Interface Tap
- ieee802 — IEEE 802.2; also ISO 8802
- osi — Umbrella for all families used by OSI (for example, protosw lookup)
- x25 — CCITT X.25 in particular
- osinet — AFI = 47, IDI = 4
- gosip — U.S. Government OSI

protocol name
Identifies a protocol. This field is currently only used for the inet family. For any other family, the protocol name field contains a hyphen (–). The protocol name identifier follows the same rules as network IDs:

- Consists of non–NULL characters
- Has a length of at least 1
- Has no maximum length specified

The protocol name field may contain:

- **icmp** — Internet Control Message Protocol
- **tcp** — Transmission Control Protocol
- **udp** — User Datagram Protocol

network device
Specifies the full pathname of the device used to connect to the transport provider. Typically, this device is in the */dev* directory. The network device must be specified.

directory lookup libraries
Support a directory service (that is, a Name–to–Address Mapping service) for the network. This service is implemented by the UNIX System V Name–to–Address Mapping feature. If a network is not provided with such a library, the Name– to–Address Mapping feature does not work. A hyphen (–) in this field shows that lookup libraries, and therefore Name–to–Address Mapping, are unavailable.

directory lookup libraries
Support a directory service (that is, a Name–to–Address Mapping service) for the network. This service is implemented by the UNIX System V Name–to–Address Mapping feature. If a network is not provided with such a library, the Name– to–Address Mapping feature does not work. A hyphen (–) in this field shows that lookup libraries, and therefore Name–to–Address Mapping, are unavailable.

The *directory lookup library* field consists of a comma–separated list of full pathnames to dynamically linked libraries. Literal commas may be embedded as \, ; backslashes as \\. Lines in */etc/netconfig* that begin with a pound sign (#) in column 1 are comments.

As system administrator, you determine the *order* of the entries in the netconfig database. Since the Network Selection library routines that access *netconfig* directly return entries in order, beginning at the top of the */etc/netconfig* file, the order in which you enter networks in the file becomes the default search path for applications choosing networks to connect to.

Figure 7–2 shows a sample *netconfig* file.

```
starlan    tpi_cots     v  osinet    -   /dev/starlan   /usr/lib/straddr.so
starlandg  tpi_clts     v  osinet    -   /dev/starlandg /usr/lib/straddr.so
npack      tpi_cots     v  localnet  -   /dev/npack     /usr/lib/npack.so
tcp        tpi_cots_ord v  inet      tcp /dev/tcp       /usr/lib/tcpip.so
udp        tpi_clts     v  inet      udp /dev/udp       /usr/lib/tcpip.so
ticlts     tpi_clts     v  loopback  -   /dev/ticlts    /usr/lib/straddr.so
ticots     tpi_cots     v  loopback  -   /dev/ticots    /usr/lib/straddr.so
ticotsord  tpi_cots_ord v  loopback  -   /dev/ticotsord
```

Figure 7–2: Sample *netconfig* File

The NETPATH Environment Variable

Setting the **NETPATH** variable allows you to influence the selection of transports used by applications. The **NETPATH** environment variable is not set in */etc/profile.* You can, however, set it in a user's *$HOME/.profile.*

NETPATH, like the **PATH** variable, consists of a colon–separated list of network IDs. Each network ID corresponds to the *network ID* field of a record in the *netconfig* database.

The set of default networks is different for the routines that access *netconfig* directly and the routines that access *netconfig* via the **NETPATH** environment variable.

- For the routines that access *netconfig* directly, the set of default networks is the entire *netconfig* file.
- For the routines that access *netconfig* via **NETPATH,** the set of default networks is the visible networks in the *netconfig* file.

A network is visible if you include a **v** flag in the flag field. If **NETPATH** is unset, these visible networks are the default search path for this second group of access routines.

NETPATH is described in the **environ**(5) manual page in the *User's Reference Manual.*

Name–to–Address Mapping

The Name–to–Address Mapping feature allows an application to obtain the address of a service on a specified machine in a transport–independent manner.

Name–to–Address Mapping consists of routines for use by application programs. These routines are used to obtain addresses of services on given hosts. All routines are combined into a library, one for each transport provider. The library to use for a specific transport provider is named in the */etc/netconfig* file in the entry for each network. The **netdir_getbyname()** routine dynamically links the library named in the directory lookup libraries field of the */etc/netconfig* file.

The Name–to–Address Mapping routines are:

```
netdir_getbyname
netdir_getbyaddr
netdir_free
netdir_mergeaddr
taddr2uaddr
uaddr2taddr
netdir_options
```

These routines are described in the *UNIX SVR4 Programmer's Guide: Networking Interfaces*, on the **getnetconfig**(3N) manual page and on the **netdir**(3N) manual page.

The Name–to–Address Mapping libraries provide all the above routines. The libraries are:

tcpip.so
Contains the Name–to–Address Mapping routines for the TCP/IP protocol suite.

straddr.so
Contains the Name–to–Address Mapping routines for any protocol that accepts strings as addresses. ISO, STARLAN, and the loopback drivers are examples.

The Name–to–Address Mapping Libraries

You must create and maintain files for each of the Name–to–Address Mapping libraries.

tcpip.so.

The routines in this dynamic library create addresses from the */etc/hosts* and */etc/services* files available with the TCP/IP package. The */etc/hosts* file contains two fields, the machine's IP address and the machine name. For example:

```
192.11.108.01    bilbo
192.11.108.16    elvis
```

The */etc/services* file contains two fields, a service name and a port number with one of two protocol specifications, either tcp or udp. For example:

```
rpcbind    111/udp
rpcbind    111/tcp
login      513/tcp
listen     2766/tcp
```

For an application to use this library to request the address of a service on a particular host, the host name must appear in the */etc/hosts* file and the service name must appear in the */etc/services* file. If one or the other does not appear, the Name–to–Address mapping routines return an error.

straddr.so.

The routines in this dynamic library create addresses from files that have the same format as the **tcpip.so** library. The **straddr.so** files are */etc/net/transport/hosts* and */etc/net/transport/services*. *transport* is the local name of the transport provider that accepts string addresses (specified in the *network ID* field of the */etc/netconfig* file). For example, the host file for starlan would be */etc/net/starlan/hosts*, and the service file for starlan would be */etc/net/starlan/services*.

Even though most string addresses do not distinguish between host and service, separating the string into a host part and a service part provides consistency with other transport providers. The */etc/net/transport/hosts* file therefore contains a string that is considered to be the machine address, followed by the machine name. For example:

```
bilboaddr    bilbo
elvisaddr    elvis
frodoaddr    frodo
```

The */etc/net/transport/services* file contains a service name followed by a string identifying the service port. For example:

```
rpcbind    rpc
listen     serve
```

The routines create the full string address by combining the host address and the service port, separating the two with a dot (.).

For example, the address of the **listen** service on **bilbo** would be **bilboaddr.serve** and the address of the **rpcbind** service on **bilbo** would be **bilboaddr.rpc**.

When an application requests the address of a service on a particular host on a transport provider that uses this library, the host name must appear in */etc/net/transport/hosts* and the service name must appear in */etc/net/transport/services*. If one or the other does not appear, the Name–to–Address Mapping routines return an error.

Chapter 8 Basic Networking Utilities

This chapter contains a general description of the Basic Networking Utilities. It includes the following topics:

General Description of the Basic Networking Utilities

The Basic Networking Utilities Package (BNU) lets systems using the UNIX operating system communicate with each other and with remote terminals. These utilities range from those used to copy files between systems (**uucp** and **uuto**) to those used for remote login and command execution (**cu, ct**, and **uux**).

BNU enables you to:

- Transfer files between UNIX systems
- Perform commands on a remote system
- Send mail to users on a remote system

What Hardware Is Needed?

Before your system can communicate with other systems, you must set up the hardware to complete the communications link. The type of connection determines the type of hardware you need. Possible connections are:

- Direct link (hardwired cable)
- Telephone lines (modem)
- Local area network (LAN)
- X.25 network
- Datakit network

Direct Links

You can create a direct link to another system by running cables between the serial ports on the two machines. Direct links are useful where two systems communicate regularly and are physically close.

A direct link permits data transfer at rates as high as 38400 bits per second (bps). Although the RS–232 standard recommends that direct links be 50 feet or less, you can separate the two systems by several hundred feet as long as noise does not become a problem. If noise becomes a problem or if a greater distance is needed between the two systems, decrease the transfer rate or place a limited distance modem at each end of the connection.

Telephone Lines

Using an Automatic Call Unit (ACU) or a modem with auto–dialing capabilities, your system can communicate with other systems over standard phone lines. An ACU is an instrument (possibly a combination of hardware and software) that dials the telephone number requested by UUCP. The remote (called) system must have a telephone modem capable of answering incoming calls.

Local Area Network

You can also use a Local Area Network (LAN) as the communication medium for BNU. Once your system is established as a node on a LAN, it can contact any other system connected to the LAN.

X.25 Network

You can communicate through an X.25 network if your system has the X.25 Host PAD product installed. X.25 connections require special hardware. See your X.25 Host PAD documentation for details.

NOTE: An active system can establish a link to another system, whereas a passive system can not. When an active system calls a passive system, the process is called polling.

What is the BNU Software?

The BNU software consist of the following:

- Commands for transferring files
- Daemons
- An internal program
- A supporting database
- Administrative files

The following directories contain most of the BNU software:

/etc/uucp
: Contains the database files.

/usr/lib/uucp
: Contains the daemons, /etc/uucp and administrative programs.

/var/spool/locks
: Contains system and/or device access lock files.

/var/spool/uucp
: Stores work files, data files, and execute files for spooled transfers. This is frequently called *the spool directory.*

/var/spool/uucppublic
: The directory to and from which UUCP can copy files (on any system). This is frequently called *the public directory.*

The BNU user programs are located in */usr/bin.*

The next sections describe the software associated with BNU.

Commands

The Basic Networking Commands can be divided into two categories: user commands and administrator commands.

User Commands

The user commands for Basic Networking are located in */usr/bin*. No special permission is needed to use these programs. They are all fully described in the *Reference Manual.*

cu(1C)
Connects your system to a remote system so you can be logged in on both at the same time. This permits you to transfer files or perform permitted commands on either system without dropping the initial link.

ct(1C)
Connects your system to a remote terminal so the user of the remote terminal can log in on your system. The user of the remote terminal can call your system and request that your system call it back. In this case, the local system drops the initial link so the remote terminal's modem is available when it is called back.

setmodem(1M)
Configures locally attached intelligent modems using initialization sequences stored in the *Dialers* file. **Setmodem**(1M) can be invoked from the command line or from system initialization sripts.

uucp(1C)
Copies a file from one system to another It creates work files and data files, queues the job for transfer, and calls the **uucico** daemon, which in turn attempts to contact the remote system to complete the transfer.

uuto(1C)

Copies files from one system to a public spool directory on another system (*var/spool/uucppublic/receive*). Unlike **uucp**, which lets you copy a file to any accessible directory on the remote system, **uuto** places the file in an appropriate spool directory and tells the remote user to pick it up with **uupick**.

uupick(1)

Retrieves files placed in */var/spool/uucppublic/receive* by a **uuto** file transfer.

uux(1C)

Creates the work, data, and execute files needed to perform commands on a remote system. The work file contains the same information as work files created by **uucp** and **uuto**. The execute files contain the command string to be performed on the remote system and a list of the data files. The data files are those files required for the command execution.

uustat(1C)

Displays the status of requested transfers (**uucp, uuto**, or **uux**). It also provides you with a means of controlling queued transfers.

uulog(1C)

Displays log of **uucp** and **uuxqt** transactions.

uuglist(1C)

Displays the list of service grades available on the system to use with **–g** option of **uucp** and **uux**.

uuname(1C)

Displays names of systems known to **uucp.**

uuencode(1)

Converts a binary file into its ASCII–encoded representation that can be sent with the **mail** command.

uudecode(1)

Decodes ASCII representations of files into binary files after they have been mailed.

Administrative Commands

You can find most of the administrative commands in */usr/lib/uucp*, along with Basic Networking shell scripts. The only exception is **uulog**, which is in */usr/bin*. All of these commands are fully described in the *Reference Manual*.

You should use the **uucp** login ID only when you administer BNU because it owns the Basic Networking and spooled data files. The home directory of the **uucp** login ID is */usr/lib/uucp*. (The other Basic Networking login ID is **nuucp**, used by remote systems to access your system. Calls from **nuucp** are answered by **uucico**.)

uucleanup (1M)

Cleans up the spooled directory. It is normally executed from a shell script called **uudemon.cleanup**, which is started by **cron**.

Uutry (1M)

Tests call processing capabilities and does a moderate amount of debugging. It invokes the **uucico** daemon to establish a communication link between your system and the remote system you specify.

uucheck (1M)

Checks for the presence of BNU directories, programs, and support files. It can also check certain parts of the *Permissions* file for obvious syntactic errors.

Networking Daemons

A daemon is a routine that runs as a background process and performs a system–wide public function. The three BNU daemons transfer files and execute commands. While these daemons run automatically, you can also run them explicitly from the system prompt.

uucico

Selects the device used for the link, establishes the link to the remote system, performs the required login sequence and permission checks, transfers data and execute files (if requested), logs results, and notifies the user by **mail** of transfer completions. When the local **uucico** daemon calls a remote system, it talks to the **uucico** daemon on the remote system during the session.

After all the required files are created, the **uucp, uuto,** and **uux** programs execute the **uucico** daemon to contact the remote system. The **uusched** and **Uutry** programs also execute **uucico.**

uuxqt

Executes remote execution requests. It searches the spool directory for execute files (always named X.*file*) that have been sent from a remote system. When it finds an X.*file* file, **uuxqt** opens it to get the list of data files that are required for the execution. It then checks to see if the required data files are available and accessible. If the files are present and can be accessed, **uuxqt** checks the *Permissions* file to verify that it has permission to execute the requested command. The **uuxqt** daemon is executed by the **uudemon.hour** shell script, which is started by **cron**.

uusched

Schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** randomizes the order in which remote systems are called. **uusched** is executed by a shell script called **uudemon.hour**, which is started by **cron**.

Supporting Data Base

These files, located in */etc/uucp*, are responsible for much of the actual networking activity associated with your BNU package. You can make changes to some of these files using the menu interface. You can edit all of them manually if you wish.

Config
: Contains a list of variable parameters within BNU. The administrator can set these parameters to configure the network manually.

Devconfig
: Contains information used to configure your network connections on STARLAN or some other network provider.

Devices
: Contains information concerning the location and line speed of the automatic call units, direct links, and network devices.

Dialcodes
: Contains dial–code abbreviations that you can use in the phone number field of *System* file entries.

Dialers
: Contains character strings required to communicate with network devices, automatic calling units, and direct links.

Grades
: Defines the job grades, and the permissions associated with each job grade, that users may specify to queue to a remote system.

Limits

Defines the maximum number of simultaneous **uucicos**, **uuxqts**, and **uuscheds** permitted on your system.

Modems

Contains a list of ports, speeds, and dialer types with the **setmodem(1M)** command uses to initialize smart modems. The *Modems* file is the **setmodem(1M)** command's version of the *Devices* file.

Permissions

Defines the level of access that is granted to systems when they attempt to transfer files or execute remote commands on your system.

Poll

Defines systems that are to be polled by your system and when they are to be polled.

Sysfiles

Assigns different or multiple files to be used by **uucico** and **cu** as *Systems*, *Devices*, and *Dialers* files.

Systems

Contains information needed by the **uucico** daemon and the **cu** program to establish a link to a remote system. This information includes the name of the remote system, the name of the connecting device associated with the remote system, when the system can be reached, telephone number, login ID, and password.

Administrative Support Files

The administrative files are described below. These files are created in spool directories and lock devices, hold temporary data, or keep information about remote transfers and executions.

TM (temporary data file)

These data files are created by the Basic Networking processes under the spool directory (that is, */var/spool/uucp/X*) when a file is received from another system. The directory *X* has the same name as the remote system that is sending the file.

The names of the temporary data files have the format TM.*pid.ddd* where *pid* is a process–ID and *ddd* is a sequential three digit number starting at 0.

When the entire file is received, the TM.*pid.ddd* file is moved or copied to the path name specified in the C.*sysnxxxxx* file that caused the transmission. If processing is abnormally terminated, the TM.*pid.ddd* file may remain in the *X* directory. These files should be automatically removed by **uucleanup.**

LCK (lock file)

These lock files prevent duplicate conversations and transfers. The names have the form LCK.*system.grade* where *system* is the name of the remote system and *grade* is the *System–Job–grade* being processed. The lock file contains the process ID of the process holding the lock. This process ID remains valid as long as the process is alive.

LK (lock file)

These lock files prevent duplicate use of calling devices. The names have the form **LK.***MAJ.maj.min* where *MAJ* is the major number of the device containing the directory entry. *maj* and *min* are the major and minor numbers, respectively, of the device itself. The lock file contains the process ID of the process holding the lock. This process ID remains valid as long as the process is active.

C. (work file)

Work files are created in a spool directory when work (file transfers or remote command executions) has been queued for a remote system.

The names of work files have the format **C.***sysnxxxx* where *sys* is the name of the remote system, *n* is the ASCII character representing the grade (priority) of the work, and *xxxx* is the four digit job sequence number assigned by **uucp**. Work files contain the following information:

- Type of request, **S** (send) or **R** (receive)
- Pathname of the file to be sent or received
- Pathname of the destination or user file name
- User login name
- List of options
- Name of associated data file in the spool directory. If the **uucp –c** or **uuto –p** option was specified, a dummy name may be used.
- Mode bits of the source file
- Remote user's login name to be notified upon completion of the transfer

D. (Data file)

Data files are created when the command line specifies to copy the source file to the spool directory. The names of data files have the format D.*systmxxxxyyy* where *systm* is the first five characters in the name of the remote system and *xxxx* is a four–digit job sequence number assigned by **uucp**. The four digit job sequence number may be followed by a sub–sequence number, *yyy*, which is used when there are several **D.** files created for a work (C.) file.

P. (Checkpoint file)

A checkpoint file is created in the spool directory when processing terminates abnormally. Unlike the TM.*pid.ddd* file, it is not removed. Therefore, when the transfer session is re–established, the length of the *checkpoint* file serves as an appropriate place to restart the transfer of the file, instead of restarting from the beginning. When checkpointing is specified for a file being transferred from another system, the checkpoint file is used instead of a **TM** file. Checkpointing occurs only between two systems that have the UNIX SVR4 BNU enhancements.

The names of the checkpoint files have the format P.*systmxxxyyy* where *systm* is the first five characters in the name of the remote system, *xxxx* is a four–digit job sequence number assigned by **uucp**. The four digit job sequence number may be followed by a sub–sequence number, *yyy*, which is used when there are several **P**. files created for a work (C.) file. When the entire file is received, the P.*systmxxxyyy* file is moved to the path name specified in the C.*sysnxxxx* file (discussed above) that caused the transmission.

X. (Execute file)

Execute files are created in the spool directory prior to remote command executions. The names of **Execute** files have the format **X.***sysnxxxx* where *sys* is the name of the remote system, *n* is the character representing the grade (priority) of the work, and *xxxx* is a four digit number assigned by **uucp**. **Execute** files contain the following information:

- Requester's login and system name
- Name of file(s) required for execution
- Input filename to be used as the standard input to the command string
- System and file name to receive standard output and stderr from the command execution
- Command string
- Option lines for return status requests

Logs

The BNU commands provide eight logs, some of which are optional.

Command Log

The command log contains a record of the commands issued by users, administrators, and operators. It can help the system administrator in trouble–shooting. The full path name of the command log is */var/spool/uucp/.Admin/command.* The format of each entry is shown in Figure 8–1:

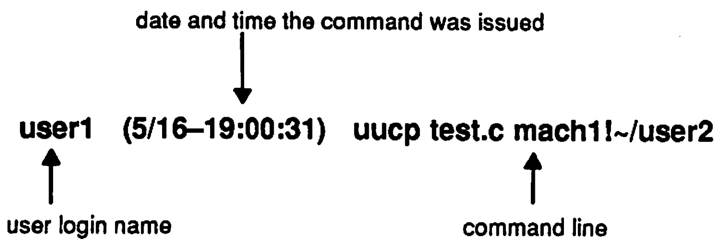


Figure 8–1: Format of the Command Log

System History Log

The system history log contains a record of each action that alters the state of the system and queue. The system history log can be generated by **uucp, uucico, uux,** or **uuxqt** programs and put into the *uucp, uucico, uux, uuxqt* subdirectories of the */var/spool/uucp/.Log* directory. Figure 8–2 shows a sample entry that **uucico** writes into */var/spool/uucp/.Log/uucico/mach1*.

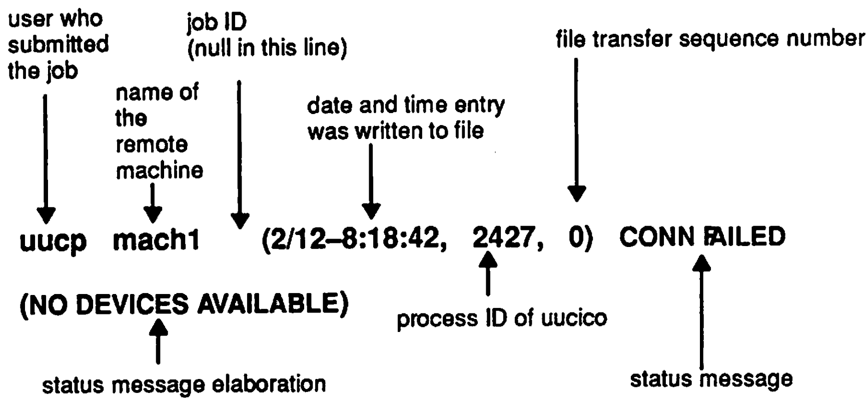


Figure 8–2: Format of System History Log

Error Log

The error log contains the error messages in the network. Error messages appear in the */var/spool/uucp/.Admin/errors* file. When the errors occur, the program aborts. In most cases, this results from file system problems. A typical entry is shown in Figure 8–3:

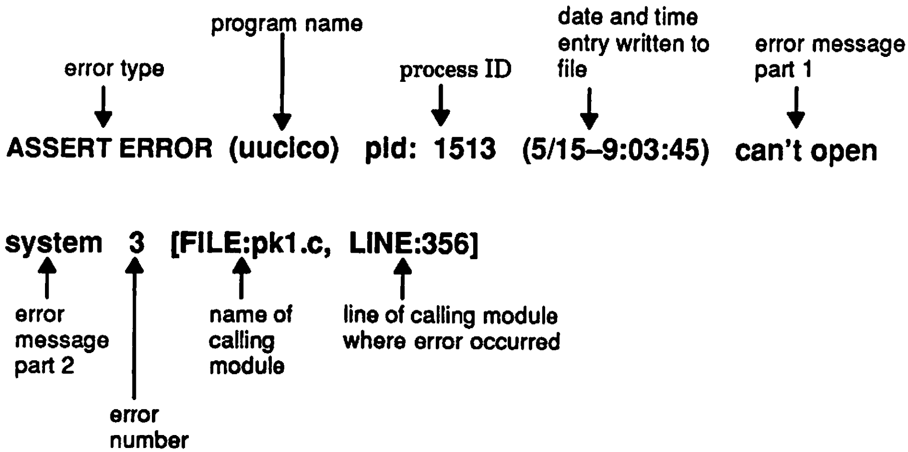


Figure 8–3: Format of Error Log

Transfer Log

The transfer log contains a record of file transfer. For example, it shows the number of bytes transferred and how long the transfer took. After both **uucicos** (master and slave) agree on the protocol, the transfer information of each file is written into */var/spool/uucp/.Admin/xferstats*. A typical entry is shown in Figure 8–4:

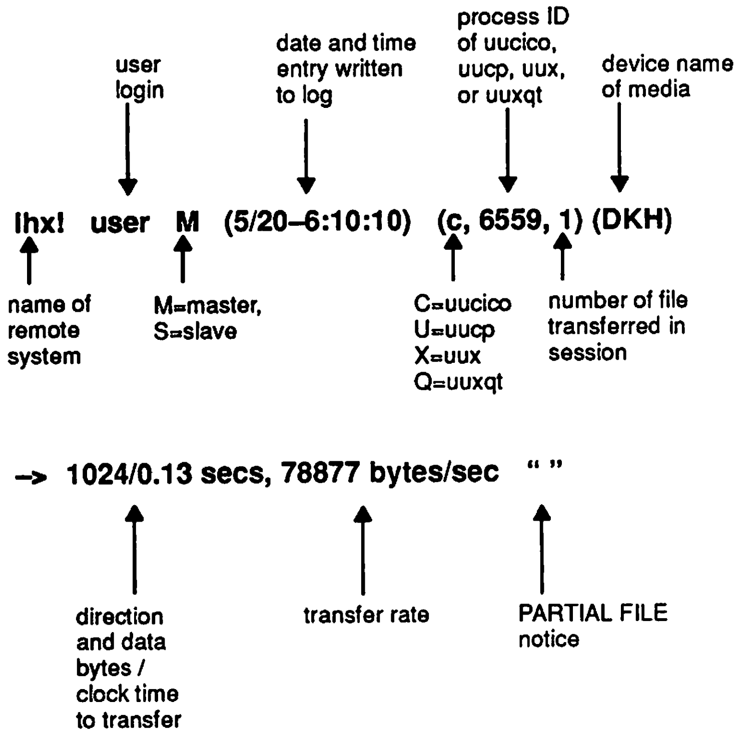


Figure 8–4: Format of Transfer Log

Report Statistics of File Transfer

By specifying the *–sfile* option with **uucp**, you can have the statistics of your file transfer reported to *file*. For each file transfer request, *file* contains three lines. For multiple file transfer requests, the size of *file* increases accordingly. A typical entry is shown in Figure 8–5:

message header	job ID	date and time transfer completed	elapsed time between creation of queue file and completion of file transfer
uucp job:	**mach1N1131**	**(6/12–13:34:58)**	**(0:0:19)**

sender node name	source file name	direction	receiver node name	destination directory/file name	requester login ID
ihnp1!	**/n15/user1/llblog**	**–>**	**mach1!**	**~/userld**	**(user1)**

message
copy succeeded

Figure 8–5: Output for File Transfer Requests

Accounting Log

The accounting log contains information needed for network charging. When a job transaction completes, accounting information is written to file */var/spool/uucp/.Admin/account*. If the job transaction is a file transfer, then accounting information is written to file */var/spool/uucp/.Admin/account* on the requesting site. If the job transaction is a remote execution, then accounting information is written to file */var/spool/uucp/.Admin/account* on the executing (target) site.

Accounting information is only collected if an account file exists and is writable by **uucp**; the file is not created automatically. As an administrator, you can create or remove the accounting log on your machine. A typical entry is shown in Figure 8–6:

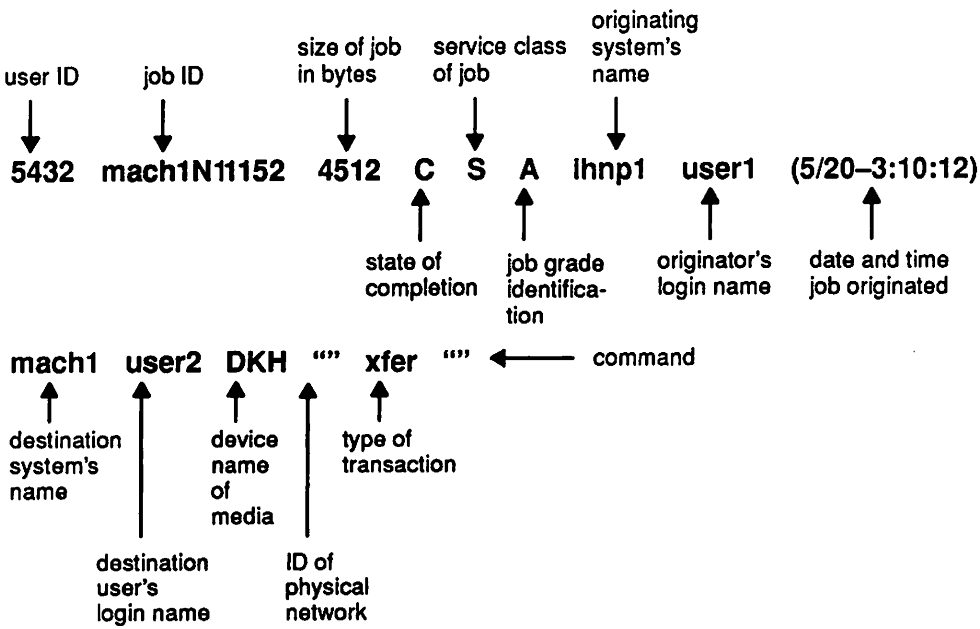


Figure 8–6: Format of Accounting Log

Security Log

The security log contains a record of the job transactions that attempt to violate system and user security measures. It is used to detect attempted security violations, such as when the requester fails to pass the security checks specified in the */etc/uucp/Permissions* file or tries to access a protected source or destination file. The occurrence is logged for further analysis in the */var/spool/uucp/.Admin/security* file. Two different entries can appear in the security log.

- *xfer* – file transfer
- *rexe* – remote execution

Figure 8–7 shows the format of the *xfer* entry in the security log, and Figure 8–8 shows the format for the *rexe* entry in the security log.

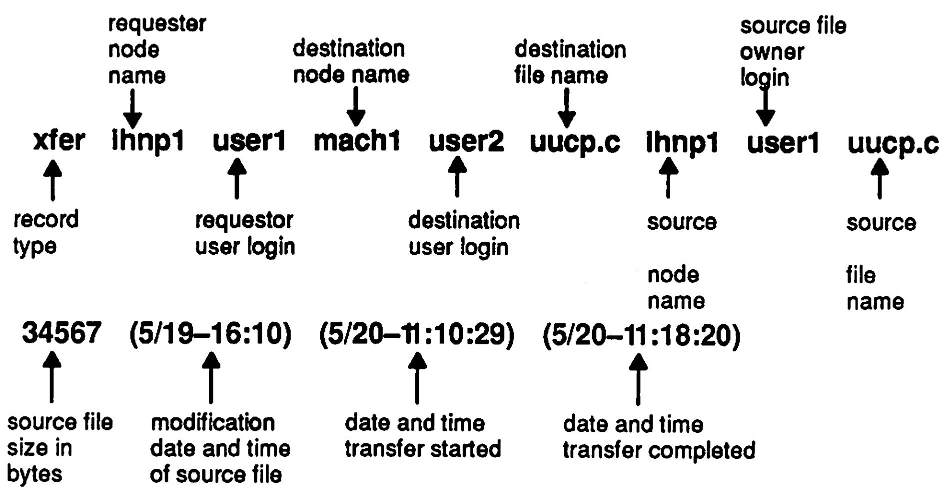


Figure 8–7: Format of *xfer* Entry in Security Log

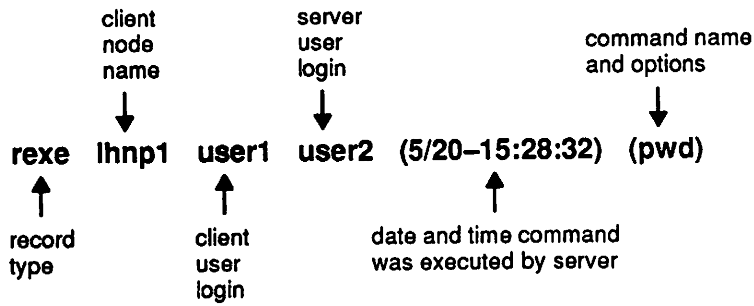


Figure 8–8: Format of *rexe* Entry in Security Log

Performance Log

The performance log contains statistics about the operation of **uucico. uucico** writes the log entries to the file */var/spool/uucp/.Admin/perflog*. Statistics are only collected if *perflog* exists when **uucico** starts; the file is not created automatically. As an administrator, you can turn on or turn off performance logging at your machine. Two types of records are written to the file; each is identified by a mnemonic type at the beginning of the record. The record types are:

- *conn* – Contains statistics about the successful establishment of a connection
- *xfer* – Contains statistics about a file transfer.

Figure 8–9 shows the performance log for the *conn* type record, and Figure 8–10 shows the performance log for the *xfer* type record.

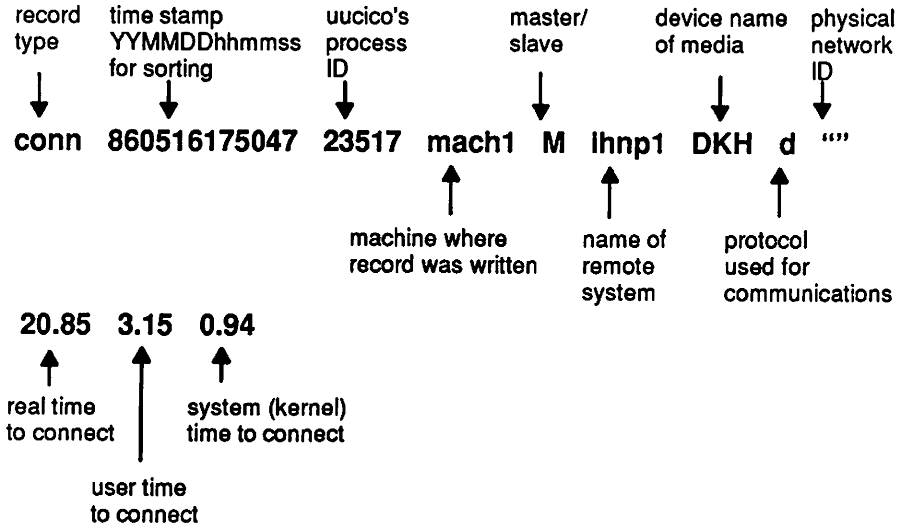


Figure 8–9: Performance Log for *conn* Type Record

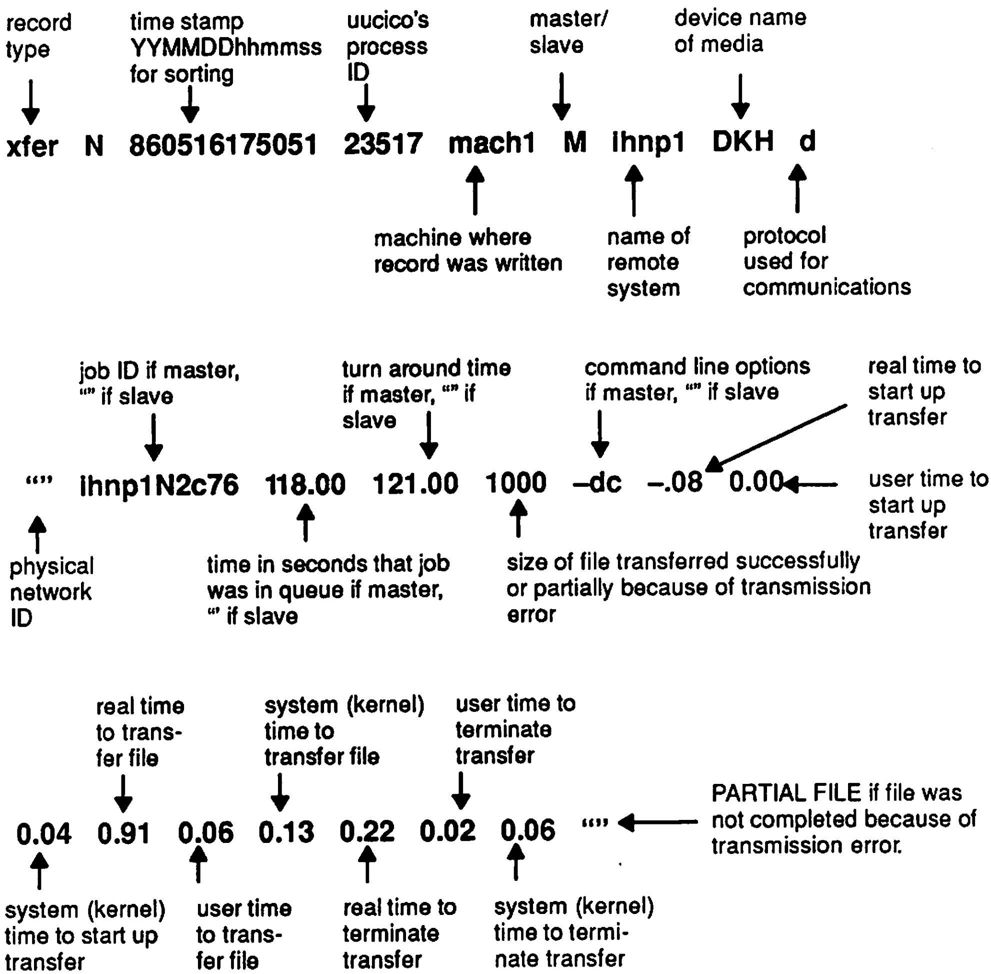


Figure 8–10: Performance Log for *xfer* Type Record

NOTE: Fields that are not applicable or unknown are marked by double quotes ("").

Foreign Log

The foreign log is a list of unknown systems that attempted to connect to the current machine. The list appears in the */var/spool/uucp/.Admin/Foreign* file. The format produced by *remote.unknown* is as shown in Figure 8–11:

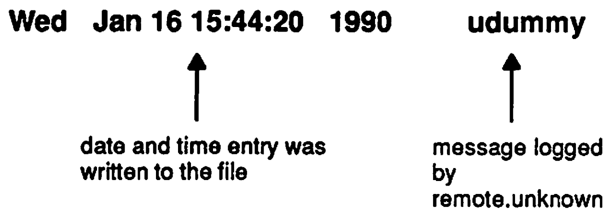


Figure 8–11: Format of Foreign Log

Chapter 9

Managing BNU

This chapter contains the following procedures for setting up, maintaining, and debugging your BNU network:

Basic Procedures

Once you understand what Basic Networking Utilities are, you need to know how to configure them for your system. You can perform the basic setup and administration through your menu interface; however, complex situations require you to enter commands at the system prompt.

To administer BNU, you must perform the following procedures:

Setup

Set up the basic networking facility and configure basic networking files. This procedure includes:

- Establishing the physical connection
- Setting the node name of the running kernel
- Identifying systems for communication
- Adding uucp logins
- Identifying devices used for communication
- Writing dialing information
- Creating access and security mechanisms
- Editing additional files
- Specifying file transfer protocols
- Verifying the link

Maintenance

Maintain your basic networking files and transactions so that your operations continue to run smoothly.

Debugging

Identify and correct common problems in basic networking operations and administration.

- Common problems
- Troubleshooting with **cu**
- Troubleshooting with **Uutry**
- Using other troubleshooting tools

Using the Menus

You can set up and administer a simple BNU using the sysadm menus.

Advantages of Using the Menus

- Online help is readily available for each selection
- You can enter information into the blanks on the screen without concern for the field order or syntax in the file(s)
- One menu selection places the necessary entries in all files
- You can set up BNU with a minimum of knowledge about the details

Disadvantages of Using the Menus

- You can not place multiple entries (with different telephone numbers or baud rates) for the same remote system name in the */etc/uucp/Systems* file.
- You can not use the menus to edit the *Dialcodes*, *Sysfiles*, and *Devconfig* files.
- You can not edit the *Dialers* file except as part of the procedure to add a connection.
- You can not edit all fields in a file through the menus. For example, you can not edit the chat script (part of the Phone field) in the *Systems* file.

NOTE: For procedures using the menu interface, see the *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface.*

Using Operating System Commands

If you prefer, you can set up and administer BNU from the system prompt using the operating system commands. The advantage to this method is that you can configure even the most complex BNU systems this way. The disadvantage is that you must know which files to edit as well as the syntax of each file you edit in order to place entries in the proper fields.

Procedure for Setting Up a Link

To set up a link for BNU, you need to accomplish the following steps. How you do so depends on the method you choose (menus vs. commands).

For menu procedures, see the *NCR UNIX SVR4 MP–RAS Administrator Guide: OA&M Menu Interface.*

The command method for each of these steps is explained in later sections of this chapter.

Step 1: Establish the physical connection.

Establish the physical link for communication between the systems (or system and modem). This book assumes the physical link is established. The next section, "Establishing the Physical Connection" contains references and suggestions for this task.

Step 2: Set the node name of the running kernel.

Give your system a name that uniquely identifies it for communication.

Step 3: Identify systems for communication.

Place entries in the *Systems* file to identify each system with which you intend to communicate. Each remote system must create an entry for your system in its *Systems* file, too.

Step 4: Identify devices used for communication.

Place entries in the *Devices* file to identify the type of link you are using (direct, modem, or LAN) and to provide other device–specific information.

Step 5: Write Dialing information.

If you are using a modem for which your *Dialers* file does not contain instructions, place the appropriate dialing information in the *Dialers* file. You can NOT do this through the menus.

Step 6: Create access and security mechanisms.

Edit the *Permissions* file to establish security mechanisms that control file system access and command execution by remote systems.

Step 7: Edit additional files.

Set up optional BNU files.

Step 8: Verify the link to be sure it works.

Test each link you set up to be sure it works correctly.

Establishing the Physical Connection

The first step in setting up a BNU link is to establish the physical connection between the two systems. This book assumes that you have already established any hardware connections required by the BNU link. Modem and cabling information can be found in the following books:

- *NCR 345x/35xx Unit Installation, Care and Cleaning*
- *NCR 345x/35xx Site Preparation Guide* (RS232 cable)
- The manual for your modem (modem connection and switch settings)

To use a modem, perform the following steps:

Step 1: Choose the setting from Figure 9–1 that accommodates the BNU software you use.

BNU Feature	Modem Setting
cu	half–duplex or full–duplex
uucp	full–duplex
cu and **uucp**	full–duplex

Figure 9–1: Modem Settings for BNU Features

Step 2: Configure a serial port to which an RS232 cable connects, specifying whether the line is used for dialing in, dialing out, or both.

Step 3: Use the **sysadm**(1M) command and select the ports menu item to set up the desired port under a port monitor. Check that the port is not currently under control of a **getty**(1M) process due to an entry in *inittab*(4).

Setting the Node Name of the Running Kernel

Each system in a BNU network must have a unique node name by which it is known to other systems in the network. Also for **mail**(1) to work correctly, the node name used by **uucp** and by **mail** must be the same.

Displaying Node Name(s)

You can use the **uuname** command to display all node names currently used by **uucp**.

- To display the names of all remote systems on your network, enter:

```
# uuname
```

- To display only the name of your system, enter:

```
# uuname -l
```

Changing the Node Name

After the initial installation of the operating system, the */etc/nodename* file contains "NodeName". You can use the **setuname** command to specify the following:

- An initial, unique node name for the running kernel on a new system
- A different node name for an older system

Restrictions

The following node name restrictions apply:

- May be up to 16 characters long, but the lock file only uses the first 7 characters, so it is best to limit the node name to 7 characters. The node name can not contain a slash (/).
- Should be unique to avoid file transfer to another system with the same name.
- Should not begin with a number. If a remote system uses the **cu**(1) command to call your system (for example, **cu** *system*) and your system name begins with a number, **cu**(1) interprets your system's name as a telephone number and no connection is made.

Syntax of setuname

```
# setuname [-s name] [-n node] [-t]
```

Common options include:

–s *name*
: Changes the system name to *name*.

–n *node*
: Changes the node name to *node*.

–t
: Temporary change. Only writes the new system or node name to the running kernel.

Refer to **uucp**(1) in the *Reference Manual* for more information about **uuname**(1M) and **setuname**(1M).

Identifying Systems for Communication

The *Systems* file (*/etc/uucp/Systems*) contains information that both the **cu** and **uucico** daemons need to establish a communication link to a remote system. Each entry in the file represents a system that can be called by your system. In addition, you can configure BNU to prevent any system that does not appear in this file from logging in on your system.

A particular system may have more than one entry in the *Systems* file. The additional entries represent alternative communication paths (or speeds) that are tried in sequential order.

Obtain the following information about each remote system from the system administrator of that system. Provide the system administrator of the remote system with the same information about your system, also.

- System node name
- Appropriate times to call
- Phone number of modem
- Login name for **uucico**
- Login password

Format of Systems File Entry

Each entry in the *Systems* file has the following format:

```
System-name Time Type Class Phone Login
```

Fields

System Name

Contains the node name of the remote system. The system name should not contain slashes (/).

Time

Specifies the day–of–week and time–of–day (optional) when the remote system can be called.

The format of the *Time* field is:

```
daytime[;retry]
```

The *day* portion may be a list containing some of the following:

- **Su Mo Tu We Th Fr Sa** — Indicates call on specific days (for example, Mo Fr)
- **Wk** — Indicates call any week–day (Mo Tu We Th Fr)
- **Any** — Indicates call any day (including weekends)
- **Never** — Indicates a passive arrangement with the remote system. Your system never initiates a call to the remote system. Instead, the remote system must initiate the call. (See the *Permissions* file.)

The *time* portion should be a range of times specified in 24–hour notation, for example 1000–1430 for "10:00 a.m. to 2:30 p.m." If you do not specify a time portion, **Any** is assumed. A time range can span 0000. For example, 0800–0600 means all times are allowed other than times between 6 a.m. and 8 a.m.

An optional subfield, *retry*, permits you to specify the minimum time (in minutes) before retrying a failed attempt. The default wait is 60 minutes. The subfield separator is a semicolon (;). The field contains no spaces between entries.

The following example specifies to call any time, but wait at least 9 minutes before retrying after a failure occurs.

```
Any;9
```

In the following example, you can call the remote system on Monday, Wednesday, and Friday from 8:00 AM to 12:00 PM, but must wait at least 5 minutes before retrying after a failure.

```
MoWeFr0800-1200;5
```

Type

Specifies the device type to use when establishing the link to the remote system. This field must match the first field of a line in the *Devices* file. If you use a file transfer protocol other than the default (g), the protocol type must appear in either this field (separated from the device type keyword by a comma) or in the *Devices* file. The following keywords may appear in this field:

- ACU — Indicates that the link to a remote system is made through an Automatic Call Unit (ACU) or modem. You can connect the modem directly to the system or indirectly through a local switch.

- *Network* — Replace *Network* with either **micom** or **develcon**, indicating that the link is established through a local switch. These two switches are the only ones that contain caller scripts in the *Dialers* file. You can use other switches if you construct caller scripts and place them in the *Dialers* file.

- *System_name* — Indicates a direct link to a particular system where *System_name* is replaced by the name of the particular system (should be same as field one).

- **Direct** — Used exclusively by **cu** to establish a direct connection to a remote system or a switch. You must have a separate entry in the *Devices* file for each dialout line referenced by the **cu -l** option. Enter the word **direct** exactly as shown.

The following example shows the keyword ACU matching the first field of a *Devices* file entry.

```
Systems:  eagle Any ACU D1200  3-2-5-1 ogin:  nuucp /
ssword:    .

Devices:  ACU  tty02  -  D1200  hayes24
```

Class

Specifies the transfer speed of the device used to establish the communication link. It may contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers (refer to the discussion of the *Class* field in description of *Devices* file). Use the keyword **Any** to indicate devices that can be used at any speed. This field must match the *Class* field in the associated *Devices* file entry as shown:

```
Systems:  eagle Any ACU D1200 3-2-5-1 ogin: nuucp \
ssword:

Devices:  ACU tty02 - D1200 hayes24
```

If information is not required for this field, use a dash (–) as a place holder for the field.

Phone

Provides the telephone number (token) of the remote system to automatic dial modems. The phone number consists of an optional alphabetic abbreviation and a numeric part.

- The abbreviation, if present, must be listed in the *Dialcodes* file.
- In this string, an equal sign (=) tells the ACU to wait for a secondary dial tone before dialing the remaining digits.
- A dash (–) in the string instructs the ACU to pause 4 seconds before dialing the next digit.

You can add extra dashes at the end of the phone number for international calls. This causes a longer delay, thus enabling the connection to complete.

If your system is connected to a local switch, you may access other systems that are connected to that switch. The *Systems* file entries for these systems do not have a telephone number in the *Phone* field. Use a dash (–) as a place holder in the field. The associated *Devices* file entry should have a **\D** at the end of the entry to ensure that this field is not translated using the *Dialcodes* file.

Login

Contains login information given as a series of fields and subfields of the format. *Expect* is the sting that is received, and *send* is the string that is sent when the *expect* string is received.

```
expect send
```

The *expect* field may be made up of subfields of the form:

```
expect[-send-expect]...
```

The *send* string is sent if the prior expect is NOT successfully read. The *expect* following the *send* is the next expected string. If no characters are initially expected from the remote system, use the null string "" in the first *expect* field. All *send* fields are terminated by ENTER unless the *send* string is terminated with a \c.

In the following example, BNU expects **ogin:**. If BNU gets **ogin:**, it goes on to the next field. If it does not get **ogin:**, it sends BREAK followed by ENTER, then looks for **ogin:** again.

```
ogin:-BREAK-ogin:
```

In the following example, BNU expects **login**. If BNU gets **login**, it goes on to the next field. If it does not get **login**, it sends nothing followed by a new line, then looks for **login** again.

```
login--login
```

Password

Specifies a password for the login name. For security purposes, you should set a password.

Dialup Password

Specifies a dialup password for the line or port used in UUCP communication.

Several escape characters cause specific actions when they are a part of a string sent during the login sequence. The escape characters in Figure 9–2 is useful when using BNU communications:

Escape Char.	Description
\N	Send or expect a null character (ASCII NUL).
\b	Send or expect a backspace character.
\c	If at the end of a string, suppress the new line that is normally sent. Ignored otherwise.
\d	Delay two seconds before sending or reading more characters.
\p	Pause for 0.25 or 0.5 seconds.
\E	Start echo checking. (From this point on, whenever a character is transmsitted, it waits for the character to be received before doing anything else.)
\e	Stop echo checking.
\M	Turn on CLOCAL flag.
\m	Turn off CLOCAL flag.
\n	Send a new line character.
\r	Send or expect a carriage return.
\s	Send or expect a space character.
\t	Send or expect a tab character.
\\	Send or expect a backslash (\) character.
EOT	Send or expect EOT new line twice.
BREAK	Send or expect a break character.
\K	Same as BREAK.
\ddd	Send the ASCII character represented by these octal digits (*ddd*).

Figure 9–2: BNU Escape Characters

General Examples of Systems File Entries

Example 1

```
xyz Any;2 ACU 1200 ACpa-555-6695 ""
ogin--ogin-EOT-ogin-- ogin- BREAK-ogin
```

This example shows a line in the *Systems* file for the remote system *xyz*. Dialing is permitted at any time (**Any**); wait 2 minutes in case of failure. **uucp** normally tries the same number twice.

For the ACU or dialing modem, **uucp** uses the first available line from the *Devices* file that has a first field containing ACU.

The phone number passed to the dialing routine is 555–6695 plus the ACpa specified in the *Dialcodes* file (probably 1–215, the area code for southeastern Pennsylvania). Pauses occur after the area code and after the exchange number.

After the dialer establishes connection, it tries the following:

- It sends a newline (without waiting, because of the initial null expect field).
- If the response contains "ogin", the dialer sends "nuucp".
- If the response does not contain "ogin", the dialer sends another newline (—).
- If that fails to bring an "ogin", it sends EOT\n twice.
- If the dialer still receives no "ogin", it sends another newline.
- If that does not bring an "ogin", the dialer sends a break.
- If the break fails to bring an "ogin", **uucp** gives up for the time being.

Example 2

```
kudzu Any kudzu 9600 - "" \r\d\r\d\r\d\r ogin: nuucp
ssword: sniglet
```

This example of a direct connection shows a line in the *Systems* file for the remote system *kudzu*. The systems may communicate at any time (**Any**). The second *kudzu* matches the first field of a line in the *Devices* file. The data transfer rate is 9600. No phone number is present because the connection is direct. The remainder of the line is the expect–send string for login.

Adding uucp logins

You may add one or more administrative logins to your system so incoming **uucp (uucico)** requests from different remote machines can be handled differently. Each remote machine should have an entry in its *Systems* file for your machine that contains the login ID and password that you add to your */etc/passwd* file.

The default in the */etc/passwd* file is shown below.

```
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp
nuucp:x:10:10:0000-uucp(0000):/var/spool/uucppublic:/u
sr/lib/uucp/uucico
```

This entry shows that a login request by **nuucp** is answered by */usr/lib/uucp/uucico*. The home directory is */var/spool/uucppublic*. The *x* indicates that the encrypted password is stored in */etc/shadow*.

Identifying Devices Used for Communication

The *Devices* file (*/etc/uucp/Devices*) contains information for all the devices that may be used to establish a link to a remote system. It contains information for modems with auto–dialing capabilities, direct links, and network connections.

This file works closely with the *Dialers*, *Systems*, and *Dialcodes* files. Become familiar with all of these files before attempting to change any of them.

NOTE: If you remove entries from the *Devices* file, you may wish to remove the corresponding terminal services using the **ports** submenu under **sysadm**.

Format of Devices File Entry

Each entry in the *Devices* file has the following format:

```
Type Line Line2 Class Dialer-Token-Pairs
```

Fields

Type

Describes the type of link, and should match the third field of the *Systems* file. If you are using **cu**(1) and issuing commands directly to the modem, the *Type* field should contain the keyword **Direct**. In general, it may contain one of the following keywords.

- **Direct** — Indicates a Direct Link to another system or a switch.

- ACU — Specifies that the link to a remote system is made through an automatic call unit (Automatic Dial Modem). This modem may be connected either directly to your system or indirectly through a Local Area Network (LAN) switch.

- *LAN_Switch* — Specifies the name of the LAN or switch. For instance, **STARLAN** could be the name for the AT&T STARLAN network. **Develcon** could be the name for a Develcon switch connection.

- *Sys–Name* — Specifies a direct link to a particular system. (Replace *Sys–Name* with the name of the system.) This naming scheme conveys that the line associated with this *Devices* entry is for a particular system in the *Systems* file.

The keyword in the *Type* field is matched against the third field of *Systems* file entry.

```
Devices:  ACU term/11 - 1200 penril

Systems: eagle Any ACU 1200 3251 ogin: nuucp ssword:
Oakgrass
```

You can designate a protocol to use for a device within this field. See the "Protocols" section at the end of the description of this file (*Devices*).

Line

Contains the device name of the line (port) associated with the *Devices* entry. For instance, if the Automatic Dial Modem for a particular entry was attached to the /dev/term/11 line, the name entered in this field would be term/11. You can use the optional modem control flag M in the Line field to indicate that the device should be opened without waiting for a carrier. For example:

```
term/11,M
```

Line2

If you use the keyword **ACU** in the Type field and the ACU is an 801 type dialer, this field should contain the device name of the 801 dialer. (801 type ACUs do not contain a modem. Therefore, a separate modem is required and should be connected to a different line, defined in the *Line* field.) One line is allocated to the modem and another to the dialer. Because the system does not normally use this type of configuration, this field is ignored, but it must still contain a dash (–) as a placeholder.

Class

If you use the keyword **ACU** or **Direct** in the *Type* field, *Class* may be just the speed of the device. However, it may contain a letter and a speed (for example, C1200, D1200) to differentiate between classes of dialers (Centrex or Dimension PBX). This distinction is necessary if you have one network dedicated to serving only internal office communications while another handles the external communications. You should distinguish between line(s) used for internal communications and those used for external. You must make the same distinction in the *Systems* file because a match is made against the fourth field of *Systems* file entries as shown below:

```
Devices:  ACU term11 - D1200 penril

Systems:  eagle Any ACU D1200 3251 ogin: nuucp \
ssword: Oakgrass
```

Some devices can be used at any speed, so the keyword **Any** may be used in the *Class* field. If you use **Any**, the line matches any speed requested in a *Systems* file entry. If this field is **Any** and the *Systems* file Class field is **Any**, the speed defaults to 1200 bps.

Dialer

Identifies the dialer in the *Dialers* file. If you enter **Direct**, a Direct entry is also generated so that the command line **cu –l** *ttynn* is supported.

Dialer–Token–Pairs

Contains pairs of dialers and tokens. The dialer portion may be the name of an automatic dial modem, a LAN switch, or it may be **direct** or **uudirect** for a Direct Link device. You can have any number of Dialer–Token–Pairs. The token portion may be supplied immediately following the dialer portion, or, if not present, taken from a related entry in the *Systems* file.

This field has the format *dialer token [dialer token]* where the last pair may or may not be present, depending on the associated device (dialer). In most cases, the last pair contains only a dialer portion and the token is retrieved from the Phone field of the *Systems* file.

A valid entry in the dialer portion may be defined in the *Dialers* file or may be one of several special dialer types. These special dialer types are compiled into the software and are therefore available without having entries in the *Dialers* file.

- 801 — Bell 801 auto dialer
- TLI — Transport Level Interface Network (without STREAMS)
- TLIS — Transport Level Interface Network (with STREAMS)

These special dialer types are checked first, and then the *Dialers* file is checked for all remaining dialer fields.

The Dialer–Token–Pairs (DTP) field may be structured differently, depending on the device associated with the entry:

- If an automatic dialing modem is connected directly to a port on your system, the DTP field of the associated *Devices* file entry has one pair. This pair is normally the name of the modem. This name is used to match the particular *Devices* file entry with an entry in the *Dialers* file. Therefore, the dialer field must match the first field of a *Dialers* file entry as shown below:

  ```
  Devices: ACU term/11 - 1200 att2212c

  Dialers:  att2212c =+-, "" atzod,o12=y,o4=n\r\c
  \006 atT\T\r\c ed
  ```

  Notice that only the *dialer* portion (att2212c) is present in the DTP field of the *Devices* file entry. This means that the token to be passed on to the dialer (in this case the phone number) is taken from the Phone field of a *Systems* file entry.

- If a direct link is established to a particular system, the DTP field of the associated entry would contain the keyword **direct** or **uudirect**. This is true for both types of direct link entries, **Direct** and *System–Name* (refer to discussion on the *Type* field).

- If you wish to communicate with a system that is on the same local network switch as your system, your system must first access the switch and the switch can make the connection to the other system. In this type of entry, there is only one pair. The dialer portion is used to match a *Dialers* file entry as shown below:

  ```
  Devices:  develcon term/13 - 1200 develcon

  Dialers:  develcon "" "" \pr\ps\c est:\007 \E\D\e
  \007
  ```

As shown, the token portion is left blank, indicating that it is retrieved from the *Systems* file. The *Systems* file entry for this particular system contains the token in the *Phone* field, which is normally reserved for the telephone number of the system (refer to *Systems* file, *Phone* field). This type of DTP contains an escape character (**\D**), which ensures that the contents of the *Phone* field are interpreted as a valid entry in the *Dialcodes* file.

- If an automatic dialing modem is connected to a switch, your system must first access the switch and the switch makes the connection to the automatic dialing modem. This type of entry requires two dialer–token pairs. The dialer portion of each pair (fifth and seventh fields of entry) is used to match entries in the *Dialers* file as shown below:

```
Devices: ACU term/14 - 1200 develcon dial att2212c

Dialers: develcon " " \pr\ps\c est:\007 \E\D\e \007


Dialers: att2212c =+-, " atzod,o12=y,o4=n\r\c \006
atT\T\r\c ed
```

In the first pair, **develcon** is the dialer and **dial** is the token that is passed to the Develcon switch to tell it which device (auto dial modem) to connect to the system. This token is unique for each LAN switch since each switch can be set up differently. Once the modem is connected, the second pair is accessed, where **att2212c** is the dialer and the token is retrieved from the *Systems* file.

Two escape characters are permitted at the end of a DTP (dialer–token–pair) field:

\T

Specifies that the Phone (token) field should be translated using the *Dialcodes* file. This escape character is normally placed in the *Dialers* file for each caller script associated with an automatic dial modem (hayes, hayes24, and so on). Therefore, the translation does not take place until the caller script is accessed. If the dialer is an internal dialer, \T is the default.

\D

Indicates that the Phone (token) field should not be translated using the *Dialcodes* file. If no escape character is specified at the end of a *Devices* entry, **\D** is the default. A **\D** is also used in the *Dialers* file with entries associated with network switches (**develcon** and **micom**).

Protocols

You can choose the protocol to use with each device, or you can use the default. If you do specify the protocol, you must do so in the form *Type Protocol[(parameters)]* (for example, **STARLAN,eg**). Available protocols are:

g

Generic packet protocol. It provides error detection and retransmission intended for use over potentially noisy lines. By its nature, it is relatively slow. Two parameters characterize the **g** protocol, *windows* and *packetsize*. *windows* indicates the number of packets which may be transmitted without waiting for an acknowledgement from the remote host. *packetsize* indicates the number of data bytes in each packet. *windows* value is set at 7, and *packetsize* is set at 64 bytes.

G

Identical to the **g** protocol in that it provides the same error detection and retransmission. However, in addition, the **G** protocol allows the number of windows and the packet size to be varied to match the characteristics of the transmission medium. When properly configured, performance can be significantly better than the **g** protocol. *windows* may range from 1 to 7, and *packetsize* may range from 32 to 4096 bytes, in powers of 2 (that is, 32, 64, 128, 256, 512, 1024, 2048, 4096).

d

Is optimized for Datakit networks. Use it on Datakit networks.

e

Assumes error free transmission and performs no error checking or retransmission. Therefore it is the fastest of these protocols. It should be used for reliable local area networks. There are no parameters to be tuned within the **e** protocol.

f

Is optimized for reliable, seven–bit networks such as X.25 packet assembler/disassemblers (PAD's). The protocol has simple error checking and retransmission. The **f** protocol is available in release 2.00.02 and later of the base operating system.

The following example uses the **e** protocol over a STARLAN local area network. If the **e** protocol is not available, **g** is used.

```
STARLAN,eg starlan - - TLIS \D
```

The following example uses the **G** protocol on a high speed modem. The number of windows is set to 7, and the packetsize is 512 bytes. If the **G** protocol is unavailable, the standard **g** protocol is used.

```
ACU,G(7,512)g term/11 - 9600 att2296a
```

Presumably, seven windows with a packet size of 512 bytes provide optimum throughput for the specified device.

For incoming connections, you may specify the preferred protocol priority and parameters in the *Config* file using the Protocol parameter.

Modem Initialization

The */etc/uucp/Modems* file contains information which the **setmodem**(1M) command uses to initialize locally-attached smart modems. The *Modems* file has the same format as the *Devices* file. The *Dialer—Token–Pairs* field, however, specifies a chat sequence in the *Dialers* file used to initialize a smart modem. For example, *Modems* may contain the following entry:

```
ACU term/00,M - 9600 dsi962init \D
```

The *Dialers* file may contain the following entry:

```
dsi9624init =,-, "" \M\d\d\dAT&FE0K1Q1V1&C1&D0*F3*d0&W
```

You may then invoke **setmodem**(1M) either interactively or from */etc/uucp/init.d*.

Writing Dialing Information

The *Dialers* file (*/etc/uucp/Dialers*) specifies the initial conversation (handshaking) that must take place on a line before it can be made available for transferring data. This conversation is usually a sequence of transmitted/expected ASCII strings that is often used by an ASCII dialer (such as the ACU) to dial a phone number.

Format of Dialers File

Each entry in the *Dialers* file has the following format:

```
Dialer Substitutions Expect-send [Expect-send ...]
```

Fields

Dialer
Matches the fifth and additional odd numbered fields in the *Devices* file. If a match succeeds, the *Dialers* entry is interpreted to perform the dialer negotiations. *Dialer* names ending in "init" are conventially used by the **setmodem**(1M) command. These names are referenced in the *Modems* file and are used for modem initialization instead of for dialing calls.

Substitutions
Is a translate string. The first of each pair of characters is mapped to the second character in the pair. Usually the mapping translates = and – into whatever the dialer requires for "wait for dialtone" and "pause."

Expect–send
Contains character strings often called the chat script. The chat script defines the login conversation that takes place between two systems.

The following example shows some character strings distributed with BNU in the *Dialers* file. You can edit the file using a system editor, such as **vi**.

```
penril =W-P "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c
       -) y\c : \E\TP > 9\c OK
ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
vadic =K-K "" \005\p *-\005\p-*\005\p-* D\p BER?
      \E\T\e \r\c LINE
develcon "" "" \pr\ps\c est:\007 \E\D\e \n\007
micom "" "" \s\c NAME? \D\r\c GO
direct
uudirect "" "" \r\d in:--in:
rixon =&-% "" \r\r\d $ s9\c )-W\r\ds9\c-) s\c : \T\r\c
       $ 9\c LINE
hayes =,-, "" \M\dAT\r\c OK\r \EATDT\T\r\c CONNECT
att4000 =,-, "" ATZ\r\p\p OK\r ATZ\r OK\r\c \EATDT\T\r
        \c CONNECT
att4024 =+-, "" atzod,o12=y,o4=n\r\c \006 atT\T\r\c ed
att2212c =+-, "" atzod,o12=y,o4=n\r\c \006 atT\T\r\c
          ed
att2224b =+-, "" atT\T\r\c ed
att2224ceo =+-, "" atzod,o12=y,o4=n,\\n3\\c1\\j0\\q0\\
        g0\r\c \006 atT\T\r\c Connected
att2224g =+-, "" atzod,o12=y,o4=n,o1=n\r\c \006
  atz\\n3\\c1\\j0\\q0\\g0\r\c "" \datT\T\r\c Connected
att2224 =+-,"" \r\c :--: T\T\r\c red
att2248a =+-, "" atzod,o12=y\r\c \006 atT\T\r\c
           Connected
att2296a =+-, "" atzod,o12=y,o50=y,o51=n,o55=n,o69=n\r
\c \006 atz\\n3\\c1\\j0\\q0\\g0\r\c " \datT\T\r\c
Connected
nls "" "" NLPS:000:001:1\N\c
```

Escape Characters

Figure 9–3 defines escape characters (those beginning with \) commonly used in the *Dialers* file. Additional escape characters are described with the *Systems* file.

Escape Char.	Description
\N	Send or expect a null character (ASCII NUL).
\c	If at the end of a string, suppress the new line that is normally sent. Ignored otherwise.
\d	Delay two seconds before sending or reading more characters.
\p	Pause for 0.25 or 0.5 seconds.
\E	Start echo checking. (From this point on, whenever a character is transmsitted, it waits for the character to be received before doing anything else.)
\e	Stop echo checking.
\M	Turn on CLOCAL flag.
\m	Turn off CLOCAL flag.
\n	Send a new line character.
\r	Send or expect a carriage return.
\K	Send a BREAK.
\ddd	Send the ASCII character represented by these octal digits (*ddd*).
\D	Indicates phone number or token without *Dialcodes* translation.
\T	Indicates phone number or token with *Dialcodes* translation.

Figure 9–3: Escape Characters in the Dialers File

Example

The att2212c entry in the *Dialers* file is executed as follows. First, the telephone number argument is translated, replacing any = with a **W** (wait for dialtone) and replacing any – with a **P** (pause). The handshake given by the remainder of the line works as follows:

""

Wait for nothing. (In other words, proceed to the *expect–send* string.)

=+–

Secondary dial tone and pause.

atzod

Enter command mode, reset modem, set options to default.

o12=y

Set option 12 to 'y' (transparent data mode).

o4=n\r\c

Set option 4 to 'n' (do not disconnect on received spaces), terminate with a carriage return but no newline.

\006

Wait for acknowledge signal (ACK).

atT\T\r\c

Enter command mode. Use tone dialing. Translate the phone number and terminate with a carriage return, but no newline.

ed

Expect "ed" (answered).

Adjustments for International Calls

International calls require more than the default time to make a uucp connection. You can modify either the *Dialers* or the *Systems* file to handle this situation, but the *Systems* file is the preferred place. Why make all calls on a modem delay if only a few specific systems make international calls?

To modify the *Dialers* file to delay the time when the system begins looking for a connect, add extra \d entries. For example, changing

```
hayes =,- "" \M\dAT\r\c OK\r \EATDT\r\c CONNECT \m\c
```

to

```
hayes =,-, "" \M\dAT\r\c OK\r \EATDT\T\r\d\d\d\d\c CONNECT \m\c
```

causes the system to delay for 8 seconds before looking for a connect (each \d causes a 2 second delay).

Creating Access and Security Mechanisms

The *Permissions* file (*/etc/uucp/Permissions*) specifies the permissions that remote systems have with respect to login, file access, and command execution. Options are provided for restricting the remote system's ability to request files and to receive files queued by the local site. Another option is available to specify the commands that a remote site can execute on the local system.

Format of the Permissions File

Each entry is a logical line. If a logical line contains more than one physical line, each physical line except the last ends with a back slash (\) to indicate that the entry continues on the next line. Within an entry, white space separates the options. Each option is a *name=value* pair; that is, an option *name* followed by an equal sign (=) and the value. No white space is permitted within an option assignment.

Comment lines begin with a pound sign (#) and they occupy the entire line up to a newline character. Blank lines are ignored (even within multi–line entries).

There are two types of *Permissions* file entries:

LOGNAME
: Specifies the permissions that take effect when a remote system logs in on (calls) your system.

MACHINE
: Specifies permissions that take effect when your system logs in on (calls) a remote system. You should have a MACHINE=*local system name* entry for the local system to make sure **uux**(1) works on the local system.

Considerations

Consider the following when using the *Permissions* file to restrict the level of access granted to remote systems:

- All login IDs used by remote systems to login for BNU communications must appear in one and only one LOGNAME entry.
- Any site that is called whose name does not appear in a MACHINE entry has the following default permissions/ restrictions:
  - Local send and receive requests are executed.
  - The remote system can send files to the */var/spool/uucppublic* directory on your system.
  - The commands sent by the remote system for execution on your system must be one of the default commands; usually **rmail**.
- Unless you have a unique login and password for a remote system that calls you, you do not know if the system is who it says it is.
- In the case of LAN connections, where no actual login takes place, a LOGNAME entry must exist for the userid under which the server (*/usr/bin/server*) runs. This entry is usually root.

Options

REQUEST

When a remote system calls your system and requests to receive a file, this request can be granted or denied. The REQUEST option specifies whether the remote system can request to set up file transfers from your system.

To permit the remote system to receive a requested file, specify:

```
REQUEST=yes
```

To prevent a remote system from receiving a requested file, specify:

```
REQUEST=no
```

If you do not specify a REQUEST option, the default (REQUEST=no) is used. The REQUEST option can appear in either a LOGNAME entry (remote calls you) or a MACHINE entry (you call remote).

SENDFILES

When a remote system calls your system and completes its work, it may attempt to take work your system has queued for it. The SENDFILES option specifies whether your system can send the work queued for the remote system.

To permit your system to send the work that is queued for the remote system as long as the remote system is logged in as one of the names in the LOGNAME options, specify:

```
SENDFILES=yes
```

This string is mandatory if your system is in a "passive mode" with respect to the remote system.

To permit files queued in your system to be sent only when your system calls the remote system, specify:

```
SENDFILES=call
```

The default is SENDFILES=call. This option is significant only in LOGNAME entries, since MACHINE entries apply when calls are made out to remote systems. If the option is used with a MACHINE entry, it is ignored.

READ and WRITE

These options specify the various parts of the file system that **uucico** can read from or write to. You can use the READ and WRITE options with either MACHINE or LOGNAME entries.

The default for both the READ and WRITE options is the *uucppublic* directory as shown in the following strings:

```
READ=/var/spool/uucppublic
WRITE=/var/spool/uucppublic
```

The following strings specify permission to access any file that can be accessed by a local user with *other* permissions:

```
READ=/
WRITE=/
```

The value of these entries is a colon separated list of path names. The READ option is for requesting files, and the WRITE option for depositing files. One of the values must be a component of any full path name of a file coming in or going out.

To grant permission to deposit files in */usr/news* as well as the public directory, the following values would be used with the WRITE option:

```
WRITE=/var/spool/uucppublic:/usr/news
```

When using the READ and WRITE options, specify all path names because the path names are not added to the default list. For example, if the */usr/news* path name is the only one specified in a WRITE option, permission to deposit files in the public directory is denied.

NOREAD and NOWRITE

The NOREAD and NOWRITE options specify exceptions to the READ and WRITE options or defaults. You can use the NOREAD and NOWRITE options in both LOGNAME and MACHINE entries.

The following strings permit reading any file except those in the */etc* directory (and its subdirectories–remember, these are prefixes) and writing only to the default */var/spool/uucppublic* directory. NOWRITE works in the same way.

```
READ=/
NOREAD=/etc
WRITE=/var/spool/uucppublic
```

CALLBACK

The CALLBACK option in a LOGNAME entry specifies that no transaction takes place until the calling system is called back. This option is seldom used. If two systems set the option for each other, no conversation can start.

There are two examples of when you would use CALLBACK. From a security standpoint, if you call back a machine, you can be fairly certain it is the machine it says it is. If you are doing long data transmissions, you can choose the machine that will be billed for the longer call.

The following string specifies that your system must call the remote system back before any file transfers take place.

```
CALLBACK=yes
```

The default for the CALLBACK option is:

```
CALLBACK=no
```

COMMANDS

NOTE: The COMMANDS option can compromise the security of your system. Use it with extreme care.

The **uux** program generates remote execution requests and queues them to be transferred to the remote system. Files and a command are sent to the target system for remote execution. The COMMANDS option can be used in MACHINE entries to specify the commands that a remote system can execute on your system.

For example, the following string indicates that **rmail** is the default command a remote system may execute on your system:

```
COMMANDS=rmail
```

If a command string is used in a MACHINE entry, the default commands are overridden.

For example, the following entry overrides the COMMANDS default so that the command list for systems owl, raven, hawk, and dove now consists of **rmail, rnews,** and **lp** on your system.

```
MACHINE=owl:raven:hawk:c

COMMANDS=rmail:rnews:lp
```

You can also use full path names of commands. For example,

```
COMMANDS=rmail:/usr/lbin/rnews:/usr/local/lp
```

specifies that the command **rmail** uses the default path. The default path for remote execution is */usr/bin*. When the remote system specifies **rnews** or */usr/lbin/rnews* for the command to be executed, */usr/lbin/rnews* is executed regardless of the default path. Likewise, */usr/local/lp* is the **lp** command that is executed.

If you include **ALL** in the list, any command from the remote system(s) specified in the entry is executed. Using **ALL** gives the remote system full access to your system.

The following string illustrates two points:

```
COMMANDS=/usr/lbin/rnews:ALL:/usr/local/lp
```

- The **ALL** value can appear anywhere in the string.
- The path names specified for **rnews** and **lp** are used (instead of the default) if the requested command does not contain the full path names for **rnews** or **lp**.

If possible, use the VALIDATE option with the COMMANDS option whenever the COMMANDS option contains potentially dangerous commands like **cat** and **uucp**.Any command that reads or writes files is potentially dangerous to local security when issued by the uucp remote execution daemon (**uuxqt**).

VALIDATE

Use the VALIDATE option in conjunction with the COMMANDS option when specifying commands that are potentially dangerous to your system security. It verifies the caller's identity by requiring privileged systems to have a unique login/password for uucp transactions. Be sure to protect the login/password associated with this entry. If an outsider gets that information, that particular VALIDATE option is no longer secure.

Giving a remote system a special login and password with file access and remote execution capability is like giving anyone on that system a normal login and password on your system. If you can not trust someone on the remote system, do not grant that system a privileged login and password.

The following LOGNAME entry specifies that if a remote systems that claims to be eaglc, owl, or hawk logs in to your system, it must have used the login **uucpfriend**. If an outsider obtains the **uucpfriend** login/password, masquerading is trivial

```
LOGNAME=uucpfriendVALIDATE=eagle:owl:hawk
```

But what does this have to do with the COMMANDS option, which only appears in MACHINE entries? It links the MACHINE entry (and COMMANDS option) with a LOGNAME entry associated with a privileged login. This link is needed because the execution daemon is not running while the remote system is logged in. In fact, it is an asynchronous process with no knowledge of what system sent the execution request. Therefore, the real question is: "How does your system know where the execution files came from?"

Each remote system has its own spool directory on your system. These spool directories have write permission given only to the UUCP family of programs. The execution files from the remote system are put in its spool directory after being transferred to your system. When the **uuxqt** daemon runs, it can use the spool directory name to find the MACHINE entry in the *Permissions* file and get the COMMANDS list, or if the system name does not appear in the *Permissions* file, it uses the default list.

The following example shows the relationship between the MACHINE and LOGNAME entries:

```
MACHINE=eagle:owl:hawk \
REQUEST=yes \
COMMANDS=ALL \
READ=/ \
WRITE=/

LOGNAME=uucpz \
VALIDATE=eagle:owl:hawk \
REQUEST=yes
SENDFILES=yes \
READ=/ \
WRITE=/
```

These entries provide unlimited read, write, and command execution for the remote systems eagle, owl, and hawk. The ALL value in the COMMANDS option means that any command can be issued by any of these systems.

Using the ALL value gives the remote system unlimited access to your system. In fact, files that are only readable or writable by user **uucp** (like *Systems* or *Devices*) can be accessed using commands like **ed**. This means a user on one of the privileged systems can write in the *Systems* file as well as read it!

In the first entry, you must make the assumption that when you want to call one of the systems listed, you are really calling the systems eagle, owl, or hawk. Therefore, any files put into one of the eagle, owl, or hawk spool directories is put there by one of those systems. If a remote system logs in and says that it is one of these three systems, its execution files are also put in the privileged spool directory. You therefore have to validate that the system has the privileged login **uucpz**.

MACHINE Entry for Other Systems

You may want to specify different option values for the systems your system calls that are not mentioned in specific MACHINE entries. This may occur when there are many systems calling in and the command set changes from time to time. The name OTHER for the system name is used for this entry as shown below:

```
MACHINE=OTHER \
COMMANDS=rmail:rnews:/usr/lbin/Photo:/usr/lbin/xp
```

All other options available for the MACHINE entry may also be set for the systems that are not mentioned in other MACHINE entries.

Combining MACHINE and LOGNAME Entries

It is possible to combine MACHINE and LOGNAME entries into a single entry where the common options are the same. For example, the following entries share the same REQUEST, READ, and WRITE options.

```
MACHINE=eagle:owl:hawk \
REQUEST=yes \
READ=/
WRITE=/

LOGNAME=uucpz \
REQUEST=yes \
SENDFILES=yes \
READ=/
WRITE=/
```

You can merge these two entries as shown below:

```
MACHINE=eagle:owl:hawk \
REQUEST=yes \
LOGNAME=uucpz \
SENDFILES=yes \
READ=/  \
WRITE=/
```

Permissions File Examples

Example 1

This example represents the most restrictive access to your system. The following entry starts communications with a remote system, permitting files to be transferred only to the */var/spool/uucppublic* directory.

```
LOGNAME=nuucp
```

In this case login **nuucp** has all of the default permissions and restrictions, and can only perform default commands.

These restrictions are:

- The remote system can send files only to **uucppublic**.
- The remote system can not request to receive files (REQUEST option).
- No files queued for the remote system are transferred during the current session (SENDFILES option).
- The only commands that can be executed are defaults.

Example 2

This example is also for remote systems that log in, but places fewer restrictions. Do not distribute the login and password corresponding to this entry to the general public. Reserve it for closely coupled systems where the *Systems* file information can be tightly controlled.

```
LOGNAME=uucpz \
REQUEST=yes \
SENDFILES=yes \
READ=/ \
WRITE=/
```

This entry places the following permissions and restrictions on a system that logs in as **uucpz**:

- Files can be requested from your system (REQUEST option).
- Files can be transferred to any directory or any file that is writable by user **other**. That is a file/directory that is writable by a local user with neither owner nor group permissions (WRITE option).
- Any files readable by user **other** can be requested (READ option).
- Any requests queued for the remote system are executed during the current session. These are files destined for the system that has called in (SENDFILES option).
- The commands sent for execution on the local system must be in the default set.

Example 3

The two previous examples referred to remote systems that log in to your system. This example is a *Permissions* file entry used when calling a remote system.

```
MACHINE=eagle:owl:hawk:raven \
REQUEST=yes \
READ=/ \
WRITE=/
```

When calling any system in the MACHINE list, the following permissions prevail:

- The remote system can both request and send files (REQUEST option).
- The source or destination of the files on the local system can be anywhere in the file system.
- The only commands that are executed for the remote system are those in the default set.

Any site called that does not have its name in a system entry has the default permissions stated in Example 1, with the exception that files queued for that system are sent (the SENDFILES option is only interpreted in the LOGNAME entry).

Editing Additional Files

Dialcodes File

The *Dialcodes* file (*/etc/uucp/Dialcodes*) contains the dial–code abbreviations that can be used in the *Phone* field of the *Systems* file. To change this file, use one of the system editors (ed or vi).

Format of Dialcodes File Entry

Each entry in the file has the format shown here. The fields are defined in the paragraphs that follow.

```
abb dial-seq
```

Fields

abb

The abbreviation used in the *Systems* file Phone field.

dial–seq

The dial sequence that is passed to the dialer when that particular *Systems* file entry is accessed.

Example

The following entry works with a Phone field in the *Systems* file (for example, jt7867). When the entry containing jt7867 is encountered, the sequence 9=847–7867 is sent to the dialer.

```
jt 9=847-
```

The Poll File

The *Poll* file (*/etc/uucp/Poll*) contains information for polling specified systems. Each entry in the file contains the name of the remote system to call, followed by a TAB character and the hours the system should be called.

Format of Poll File Entry

Each entry has the format shown here. The fields are defined in the paragraphs that follow.

```
System_name     Hours
```

Fields

System_name

Contains the node name of the remote system. The system name should not contain any slashes (/).

Hours

Contains the hours that the system should be called. Separate the hours with a space.

Example

This example entry polls the system eagle every four hours.

```
eagle           0       4       8       12      16      20
```

The **uudemon.poll** script does not actually perform the poll. It merely sets up a polling work (*C.*) file in the spool directory that is seen by the scheduler. **uudemon.hour** starts the scheduler. Refer to the discussion on **uudemon.poll.**

The Sysfiles File

The */etc/uucp/Sysfiles* file permits you to provide different files for use by **uucp** and **cu** as *Systems*, *Devices*, and *Dialers* files. The following cases illustrate the usefulness of this option.

- Different *Systems* files permit login services requests and **uucp** services requests to be made to different addresses.
- Different *Dialers* files use different handshaking for **cu** and **uucp**.
- The *Systems* file in particular may become large, making it more convenient to split it into several smaller files.

Format of Sysfiles File Entry

Each entry in the *Sysfiles* file has the format shown here. The fields are defined in the paragraphs that follow.

```
service=w  systems=x:x dialers=y:y devices=z:z
```

Fields

service=*w*
: Indicates what program(s) should use the specified files. Replace *w* by **uucico, cu,** or both (separated by a colon).

systems=*x*:*x*
: Replaces *x* with the name of one or more files to be used as the *Systems* file. File names are separated by colons and read in the order presented.

dialers=*y*:*y*
: Replaces *y* with the name of one or more files to be used as the *Dialers* file. File names are separated by colons and read in the order presented.

devices=*z*:*z*
: Replaces *z* with the name of one or more files to be used as the *Devices* file. File names are separated by colons and read in the order presented.

Example

The following example uses a local *Systems* file in addition to the usual *Systems* file:

```
service=uucico:cu    systems=Systems:Local_Systems
```

The entry in */etc/uucp/Sysfiles* causes both **uucico** and **cu** to look first in */etc/uucp/Systems*. If the system they are calling does not have an entry in that file, or if the entries in the file fail, then they look in */etc/uucp/Local_Systems*.

When you define different *Systems* files for **uucico** and **cu** services, your system stores two different lists of systems. Use the **uuname** command with no option to print the **uucico** list, and the **uuname** command with the **–c** option to print the **cu** list.

The Devconfig File

The */etc/uucp/Devconfig* file is only used if you are using BNU over an AT&T STARLAN NETWORK or some other Streams–based provider. If you are using an AT&T STARLAN NETWORK, the two entries shown in the *Devconfig* file are all you need in this file.

You must also create an entry for STARLAN in your *Devices* file. Descriptions in the *Devices* file tell how to define Transport Interface devices.

The *Devconfig* file cannot be modified with the sysadm menu interface. If you want to change the contents of this file, use a text editor.

Format of Devconfig File Entry

Each entry in the *Devconfig* file has the format shown here. The fields are defined in the paragraphs that follow.

```
service=x device=y push=z[:z ...]
```

Fields

service=*x*

Specifies type of service. *x* can be **cu, uucico**, or both separated by a colon.

device=*y*

Specifies the name of a network. *y* must match an entry in the *Devices* file.

push=*z[:z ...]*

Replaces *z* by the names of STREAMS modules in the order that they are to be pushed onto the Stream. Different modules and devices can be defined for **cu** and **uucp** services.

You can modify this file to include the entries you desire.

Example

The following entries are most commonly used in the file:

```
service=cu       device=STARLAN    push=ntty:tirdwr
service=uucico   device=STARLAN    push=ntty:tirdwr
```

This example pushes ntty, then tirdwr.

The Limits File

The */etc/uucp/Limits* file limits the maximum number of simultaneous **uucicos, uuxqts,** and **uuscheds** that are running on your machine.

The *Limits* file can not be modified with the System Administration Menus command **sysadm.** If you want to change the contents of this file, use one of the text editors.

You can modify this file to include the entries you desire.

Format of Limits File Entry

Each entry in the *Limits* file has the format shown here. The fields are defined in the paragraphs that follow.

```
service=x  max=y
```

Fields

service=*x*
Specifies type of service. *x* can be **uucico, uuxqt** or **uusched.**

max=*y*
Specifies the limit that is permitted for that service.

The fields are order insensitive and lower case.

Example

The following entries should most commonly be used in the file:

```
service=uucico max=5
service=uuxqt max=5
service=uusched max=2
```

The example allows five **uucicos,** five **uuxqts,** and two **uuscheds** running on your machine.

The service invoked compares the value in the *Limits* file to the number of LCK locks in the */var/spool/locks* directory to determine whether the maximum has been reached. If there are no LCK locks and you invoke the service explicitly, no LCK lock is created since there is not much overhead for simply invoking the service. Therefore, the *Limits* file may not be enforced.

Grades File

The *Grades* file (*/etc/uucp/Grades*) contains the definitions for the job grades that may be used to queue jobs to a remote system. It also contains the permissions for each job grade. Each entry in this file represents a definition of an administrator defined job grade that allows users to queue jobs.

Format of Grades File Entry

Each entry in the *Grades* file has the format shown here. The fields are defined in the paragraphs that follow.

```
User-job-grade System-job-grade Job-size Permit-type
Id-list
```

Fields

Each entry in this file contains fields that are separated by white space. The last field in the entry is made up of sub-fields that are also separated by white space. If an entry takes up more than one physical line, then a backslash (\) is used to continue the entry onto the following line. Comment lines begin with a pound sign (#) and occupy the entire line. Blank lines are always ignored. Here is a description of each field:

User–job–grade
: Contains an administrative defined user job grade name of up to 64 characters.

System–job–grade
: Contains a one character job grade to which *User–job–grade* is mapped. The valid list of characters is A–Z,a–z, with A having the highest priority and z the lowest.

Job–size

Specifies the maximum job size that can be entered in the queue. Job–size is measured in bytes and may be a list of the following:

- *nnnn* — an integer that specifies the maximum job size for this job grade.
- *n*K — where *n* is a decimal number that represents the number of kilobytes and K is an abbreviation for kilobyte.
- nM — where *n* is a decimal number that represents the number of megabytes and M is an abbreviation for megabyte.
- Any — a keyword to specify that there is no maximum job size.

Here are some examples:

- 5000 — represents 5000 bytes
- 10K — represents 10 kilobytes
- 2M — represents 2 megabytes

Permit–type

Contains a keyword that denotes how to interpret the ID list. The following list contains the keywords and their meanings:

- **User** — ID list contains the login names of users permitted to use this job grade.
- **Non–user** — ID list contains the login names of users not permitted to use this job grade.
- **Group** — ID list contains the group names whose members are permitted to use this group.
- **Non–group** — ID list contains the group names whose members are not permitted to use this job grade.

Id–list

Contains a list of login names or group names that are to be permitted or denied queuing to this job grade. The list of names are separated by white space and terminated by a newline character. The keyword **Any** denotes that anyone is permitted to queue to this job grade.

The user job grade may be bound to more than one system job grade. The *Grades* file is searched sequentially for occurrences of a user job grade. Therefore, you should list any multiple occurrences of a system job grade should be listed according to the restriction on the maximum job size.

While there is no maximum number for the user job grades, the maximum number of system job grades allowed is 52. The reason is that more than one User–job–grade can be mapped to a System–job–grade, but each User–job–grade job grade must be on a separate line in the *Grades* file.

Example

```
mail       N    Any   User    Any
netnews    N    Any   User    Any
```

Given this configuration in a *Grades* file, these two User–job–grade grades share the same System–job–grade. Since the permissions for a Job–grade are associated with a User–job–grade and not a System–job–grade, it is even possible for two User– job–grades to share the same System–job–grades and have two different sets of permissions for each one.

Default Grade

You can define the binding of a default User–job–grade to a system job grade. Use the keyword **default** as user job grade in the User–job–grade field of the Grades file and the system job grade that it is bound to. Define the Restrictions and id fields as **Any** so that any user and any size job can be queued to this grade.

```
default     a    Any          User       Any
```

If you do not define the default user job grade, then the built–in default grade, Z, is used. Because it is assumed that the restriction field is **Any**, multiple occurrences of the default grade are not checked.

remote.unknown File

The *remote.unknown* file is a binary program that runs when a machine not found in any of the *Systems* files starts a conversation. It logs the conversation attempt and drops the connection.

Caution

If you change the permissions of the *remote.unknown* file so it can not run, your system accepts connections from any system.

The inittab File

It is recommended that you use **ttymon** for tty management instead of **getty**. If you use **getty** instead of **ttymon** and you establish a direct link between your system and another system, you must edit the *inittab* file (*/etc/inittab*) to reflect this connection. However, you MUST use **ttymon** for bidirectional traffic.

If a direct link connects your system with a system that does not have the newer version of BNU, the *inittab* file is set up differently on each system. On one system, the *inittab* file permits incoming traffic on the line; on the other system, the *inittab* file permits outgoing traffic on the line.

An inittab entry respawns a **getty** for incoming traffic, or has respawn turned **off** for outgoing traffic. For this type of link to work, one of the systems must be set up to poll the other. If the remote system permits only incoming traffic, your system must poll the remote system. If the remote system permits only outgoing traffic, the remote system must poll your system.

For the system waiting to be polled, the inittab entry must be set up to respawn getty:

```
t02:1:respawn:/etc/getty term/s02  9600NP
```

For the polling system, the */etc/inittab* entry must be set up not to respawn:

```
t01:1:off:/etc/getty term/01  un
```

The following information is required in the */etc/inittab* entry:

- Port name you want to modify
- Direction of traffic on port. If traffic is incoming only (the system is waiting to be polled), use the following entry:

  ```
  respawn:/etc/getty
  ```

  If traffic is outgoing only (the system is doing the polling), use the following entry:

  ```
  off:/etc/getty
  ```

- Transmission speed of the link (**300, 1200, 2400, 4800, 9600**, or **19200**)
- A terminal type is required so that the system administrator interface terminal listing is correct. The **un** value indicates an unknown terminal type.

Specifying File Transfer Protocols

The following protocols are available for file transfer:

- protocol g
- protocol f
- protocol e

Because it is the protocol that allows file transfer to most other systems, the g protocol is the default. The calling system determines the protocol and, after logging in to the remote system, informs the remote system of the protocol it intends to use. If the calling system specifies a protocol that the remote system does not support, start–up fails and no file transfer takes place.

g Protocol

This protocol is the most reliable and is the default protocol. Acknowledgements are performed at the packet level. The packet size is 64 bytes and the window size is 3. An 8 bit data path is used by the g protocol. You can use the g protocol without adding anything to the *Systems* or *Devices* file.

f Protocol

This protocol is designed for 7 bit ASCII data paths with control character restrictions and is less reliable than the g protocol. Binary files are converted to ASCII before transmission and then back to the original state after transmission. Checksum error checking is performed after an entire file is transferred. If the checksum is incorrect, the entire file must transmitted again. For this reason, throughput may be slower for binary files on noisy channels. If you compare the throughput of the **f** and **g** protocols when transferring an ASCII file on a known reliable link, the f protocol produces a faster transfer rate.

e Protocol

Protocol e assumes an 8 bit data path that is completely reliable. No error checking is performed. Before protocol e sends a file, it sends the receiving system a character count of the number of characters to be sent. Data is transmitted in 1024 byte blocks. This protocol gives the highest throughput with the greatest risk of undetected file corruption.

If transfer rate is not the primary consideration, use the default protocol (g) because of its reliability and compatibility with other BNU systems.

To specify another protocol, modify either the *Systems* or the *Devices* file. It is usually better to modify the *Systems* file because this permits you to single out a particular system to use a different protocol. When modifying the *Systems* file, add the protocol type to the third field (type) by separating the type keyword from the protocol type with a comma.

Examples of File Transfer Protocols

Example 1

This example specifies using protocol f whenever calling system *oosie*. When calling system *tecpubs*, protocol e is used no matter what modem is available at the time.

```
oosie   Any   oosie,f   9600   -   ""   \r\d\r\d\r\d\r
login:   nuucp

tecpubs  Any  ACU,e  Any  =9=1-555-551212  ""  \d@\d@\d@\c
ogin-EOT- ogin-BREAK1-ogin=BREAK1-ogin-\r-ogin-\r-ogin  nuucp
```

NOTE: If the calling system specifies a protocol that the remote system does not support, start–up fails and no file transfer takes place.

If you modify the *Systems* file, do not modify the *Devices* file. For the previous example, the *Devices* file would appear similar to the following:

```
oosie    term/s01  -  9600  direct
ACU      term/s02  -  1200  hayes   \D
ACU      term/s02  -  2400  hayes24 \D
ACU      term/s03  -  1200  hayes   \D
```

Example 2

If you prefer to modify the *Devices* file, add the comma and the protocol type to the first field. The file should look similar to the following:

```
oosie,f  term/s01  -  9600  direct
ACU,e    term/s02  -  1200  hayes   \D
ACU      term/s02  -  2400  hayes24 \D
ACU      term/s03  -  1200  hayes   \D
```

Whenever device *oosie* is used, protocol f is assumed and whenever the modem on *term/s02* is used, protocol e is used.

In this case, the *Systems* file is not changed and appears as follows:

```
oosie   Any   oosie   9600   -   ""   \r\d\r\d\r\d\r
login:   nuucp

tecpubs  Any  ACU,e  Any  =9=1-555-551212  ""
\d@\d@\d@\c  ogin-EOT-
ogin-BREAK1-ogin=BREAK1-ogin-\r-ogin-\r-ogin nuucp
```

Verifying the Link To Be Sure It Works

After completing the steps to install a BNU link, you need to test the configuration to be sure it works correctly. Use the following as a guide.

Step 1: To be sure you can contact the remote system, try dialing the remote system using **cu**.

- If the *Systems* file is set up correctly, you can enter:

  ```
  $ cu remote_sys_name
  ```

- If the *Systems* file is not set up correctly, but you have a **Direct** entry in the *Devices* file, you can enter:

  ```
  $ cu -l  line
  ```

Step 2: Attempt to log in and log off of the remote system. You may have to press ENTER a few times to get a login prompt.

If the previous steps succeed, the link appears to be configured correctly.

If some of your files are set up incorrectly, either of the following may be true:

- You can not connect to the remote system, but there does not appear to be a hardware problem.
- You can connect to the remote system but do not get a login prompt.

In either case, refer to the debugging and troubleshooting sections in this chapter.

Maintaining BNU

This section identifies administrative tasks necessary to keep BNU operating smoothly. Frequently, you have a choice of performing these through **cron** or from the system prompt.

Automatic Maintenance of BNU (cron)

When the system is in multi–user mode, **cron** schedules jobs from the */var/spool/cron/crontabs/root* file. Having some tasks performed automatically makes BNU administration easier. You can still perform a particular task from the command line at any time.

BNU comes with four shell scripts:

- uudemon.poll
- uudemon.hour
- uudemon.admin
- uudemon.cleanup

These scripts poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. They should be used regularly to keep your basic networking running smoothly. Normally, they are run automatically with **cron**(1M) although you can also run them manually. You can also modify these shell scripts to meet the needs of your system.

You can disable these daemons by removing the root crontab file from the */var/spool/cron/crontabs* directory. To do this, log in as **uucp** and enter:

```
# crontab -r
```

This command line removes the root crontab file from the */var/spool/cron/crontabs* directory, thus preventing **cron** from running any of the BNU daemons.

To re–enable the BNU daemons, log in as **uucp** and enter:

```
# crontab Crontab
```

The */var/lib/uucp/Crontab* file is an exact copy of the installed */var/spool/cron/crontabs/root* file. Refer to **crontab**(1) in the *Reference Manual* for information.

uudemon.admin

This shell script mails status information to the BNU administrative login (**uucp**) using the **uustat**(1C) command with the **–p** and **–q** options. Refer to the **uustat**(1C) manual page for information about these options. The **uudemon.admin** shell script should be run daily by an entry in the **uucp** crontab file.

There is no default crontab entry for **uudemon.admin**. The following is recommended:

```
48 8,12,16 * * * /bin/su uucp -c
"/etc/uucp/uudemon.admin" > /dev/null
```

uudemon.cleanup

This shell script cleans up the BNU log files and directories. Log files for individual systems are taken from the */var/spool/uucp/.Log* directory, merged, and placed in the */var/spool/uucp/.Old* directory along with older log information. Files and directories that are no longer needed in the spool directories are removed. After clean up, the script mails a summary of the status information gathered during the current day to the BNU administrative login (**uucp**).

It should be run by an entry in the **uucp** crontab file. The frequency depends on the amount of BNU traffic on your system.

There is no default crontab entry for **uudemon.cleanup**. The following is recommended:

```
45 23 * * * ulimit 5000; /usr/bin/su uucp -c \
"/usr/lib/uucp/uudemon.cleanup" \ > /dev/null 2>&1
```

uudemon.cleanup is described in detail later in this section.

uudemon.hour

The **uudemon.hour** shell script calls BNU programs on an hourly basis. It calls:

- The **uusched** program to search the spool directory for work files (*C.*) that have not been processed and schedule these files for transfer to a remote system.
- The **uuxqt** daemon to search the spool directory for execute files (*X.*) that have been transferred to your system and were not processed at the time they were transferred.

The **uudemon.hour** shell script should be run by an entry in the **uucp** crontab file. If the amount of traffic leaving and entering your system is large, it may be started once or twice an hour. If it is small, it may be started once every four hours or so.

The default root crontab entry for **uudemon.hour** is as follows:

```
41,11 * * * * /usr/lib/uucp/uudemon.hour > /dev/null
```

As delivered, this is run twice an hour. You may want it to run more often if you expect high failure rates.

uudemon.poll

The **uudemon.poll** shell script polls the remote systems listed in the *Poll* file (*/var/lib/uucp/Poll*). It creates work files (*C.*) for systems according to the entries listed in the *Poll* file.

The **uudemon.poll** script should be executed by an entry in the BNU crontab file. Schedule **uudemon.poll** to run prior to **uudemon.hour** so that the work files are present when **uudemon.hour** is called.

By default, the shell script is scheduled to run twice an hour, just before **uudemon.hour,** so that the work files are there when **uudemon.hour** is called. The default root crontab entry for **uudemon.poll** is as follows:

```
1,30 * * * * /usr/lib/uucp/uudemon.poll > /dev/null
```

Command Line Maintenance of BNU

Some maintenance is required to keep BNU files updated, ensure that the network runs properly, and track down line problems. This section describes methods to accomplish this from the command line. You may prefer to have **cron** handle most of the maintenance.

The **uustat**(1) program provides information about the latest attempts to contact various machines as well as the age and number of jobs in the queue for remote systems.

The biggest problem in a BNU network is handling the backlog of jobs that can not be transmitted to other systems. Perform the following cleanup activities routinely:

- Cleanup Undeliverable Jobs
- Cleanup the Public Area
- Combine Log Files
- Check Size of Log Files
- Cleanup Undeliverable Jobs

Cleanup Undeliverable Jobs

Regularly invoke **uustat**(1C) to display the status of connections to various systems and the size and age of the queued requests. Make sure **cron** invokes the **uudemon.admin** shell script at least once a day. This sends the administrator the current status. The following items are of particular interest:

- The age (in days) of the oldest request in each queue
- The number of times a failure has occurred when attempting to reach that system, and the reason for failure
- The age of the oldest execution request (*X.* file)

The **uudemon.cleanup** shell script is set up to remove any jobs that have been queued for several days and can not be sent. Leftover data (*D.*) and work (*C.*) files are removed after seven days, and execute (*X.*) files are removed after two days. It also provides feedback to the user indicating when jobs are not being accomplished and when these jobs are being deleted.

Cleanup the Public Area

To keep the local file system from overflowing when files are sent to the public area, **uudemon.cleanup** uses the **find** command to remove files that are older than seven days and directories that are empty. If there is not enough space for the public area, you may need to shorten this interval by changing the **uudemon.cleanup** shell script.

Combine Log Files

BNU has individual log files for each system and each program (for example, system eagle has a log file for **uucico** requests and a log file for **uuxqt** execution requests). The **uulog** program gives the user access to the information in these files by system name.

When **uudemon.cleanup** runs (usually nightly), the log files are combined and stored in a file in the */var/spool/uucp/.Old* directory. The information placed in the file the previous night is moved to another file.

By default, **uudemon.cleanup** saves files for three days prior to the current date. If space is a problem, you can edit the appropriate line in the **uudemon.cleanup** shell script to reduce the number of days the files are kept.

Check Size of Log Files

Some files may grow indirectly from uucp and other Basic Networking activities. Here are two files you should check and delete if they have become too large.

/var/adm/sulog

Keeps a history of all superuser commands. Since the **uudemon** entries in the */usr/cron/root* file use the **su** command, the *sulog* grows over time. After examining it for tampering, you should delete this file if it becomes too large.

/var/cron/log

Logs cron activities. While it grows with use, it is automatically truncated when the system goes to the multi–user state.

Common Problems

This section addresses some of the common problems you encounter with basic networking operations and administration.

Out of Space

The file system used to spool incoming or outgoing jobs can run out of space and prevent jobs from being sent or received. Not being able to receive jobs is the worse of the two conditions. When file space does become available, the system is flooded with a backlog of traffic. The shell script **uudemon.cleanup** should keep the spool directory from becoming very large. This script should be started by **cron** once a day.

Modems Connect but No Login

In an attempted connection, the modems may connect but you do not get a login prompt. Two things can cause this problem.

- The modems may be hung, in which case you should reset the modems.
- The chat script (the string of text following the phone number in the *Systems* file) may be incorrect.
- The chat script must send one or more line feed characters which will force the called system to issue its prompt.

Faulty Automatic Call Units and Modems

The automatic dial modems and/or incoming modems occasionally cause problems that make it difficult to contact other systems or receive files. These problems are usually readily identifiable since the status files accessed by **uustat** give counts and reasons for contact failure. If you suspect a bad line, you may find the **cu** command useful in trying to call another system using the suspected line.

Check for Faulty ACU/Modem

You can check if the automatic call units or modems are not working properly in several ways.

- Run **uustat –q**. This command displays counts and reasons for contact failure.

- Run **cu –d –l***line*. This command lets you call over a particular communications line and print debugging information on the attempt. If the communications line, *line*, is connected to an autodialer, you must add a telephone number at the end of the command line you execute. Otherwise, *line* must be defined as direct in the *Devices* file.

Administrative Problems

It can be very difficult to keep your *Systems* file up to date because telephone numbers, login IDs, and passwords change on remote systems. This problem can be very costly since the automatic dial modem is tied up calling a system that can not be reached. Be sure to contact the administrators of remote systems whenever you change your telephone number, login, or password, and request that they show you the same consideration.

Troubleshooting with cu

To be sure you can contact the remote system, try dialing the remote system using **cu**.

If the *Systems* file is set up correctly, you can enter:

```
$   cu   remote_sys_name
```

If the *Systems* file is not set up correctly:

- But you have a **Direct** entry in the *Devices* file, you can enter

  ```
  $   cu   -l  line
  ```

- And there is NO **Direct** entry in the *Devices* file, you can force a call on an ACU line by entering **cu** with an arbitrary number:

  ```
  $   cu   8005551212
  ```

If you get a message indicating the device is not available or the device name is unknown, your *Devices* file is not set up correctly.

If you get the message "connected", attempt to login and log off of the remote system. You may have to press ENTER a few times to get a login prompt or press BREAK to toggle baud rates.

- If you do not get a login prompt, make sure that the remote system has a **ttymon** service active on that line and the line is set up to use the same parameters.
- If you get a login prompt, try to log in (using your UUCP administrative login and password for the remote system).

If you are unable to connect to the remote system and login, use **cu** with the **–d** option to obtain diagnostic information. To test a particular line, also use the **–l** option.

Troubleshooting with Uutry

The **Uutry** program invokes **uucico** –s*systemname* (with a moderate amount of diagnostic output) to contact the specified system. The diagnostic output is placed in */tmp/systemname* and displayed on the screen. Pressing the RUBOUT or BREAK key returns the terminal to the shell while **uucico** continues to run.

The minimum retry time for a system that is busy or does not answer is 5 minutes. To invoke **Uutry** again before the retry period specified in the *Systems* file expires, you must specify the **–r** option.

The **uustat –m** output indicates if a system is busy or does not answer.

Command Format

The **Uutry** command has the following format:

```
/usr/lib/uucp/Uutry [options] systemname
```

where *systemname* is the name of the remote system to be called.

The **Uutry** command has the following *options:*

- –x*debug_level* — Overrides the default debugging level (5). The *debug_level* is a single digit (0 through 9) with higher numbers providing more debugging input.
- **–r** — Permits you to input a **Uutry** command before the retry time has expired.

Example of Uutry

If you have experienced problems in communicating with a remote system, use the **Uutry** command to aid in resolving the problem as shown below:

```
$  /usr/lib/uucp/UUtry  eagle  <CR>
conn(eagle)
Device Type ACU wanted
expect:  ("")
got it
sendthem (DELAY
^M)
expect:  (^)
^M^JAT&T ACU/Modem  ^M^J1200  BPS^M^J>got it
sendthem (PAUSE
<NO CR>)
expect:  (E)
9^M^Jsuregot it
sendthem (<NO CR>)
expect:  (:)
?  (Y?N)y^M^JNO.:got it
sendthem (ECHO CHECK ON
3P25P5P1P^MO
expect:  (^)
^M^MJ>got it
sendthem (9<NO CR)> )
expect:  (OK)
^M^JDIALING:  3251^M^JOKgot it
getto ret 5
expect:  (IN:)
^M^J^J^MJEAGLE login:got it
sendthem (^M)
expect:  (word:)
 ^M^J>nuucp^M^JPassword:got it
sendthem (^M)
Login Successful:  System-eagle
wmesg 'U'g
Protostarted g
wmesg 'h'
wmesg 'H'Y
send 00 0, Conversation Complete: Status SUCCEEDED
```

You can obtain additional diagnostic output by using the **–x** option (that is, **–x9**). For **–x9** diagnostics, the following are NOT displayed unless the person invoking **Uutry** has the userid of **root** or **uucp**:

- Phone number of the remote system
- Login name
- Password

Using Other Troubleshooting Tools

You can also use these commands to check for Basic Networking information.

uuname
Lists those machines your machine can contact.

uulog
Displays the contents of the log directories for particular hosts.

uucheck –v
Checks for the presence of files and directories needed by uucp. This command also checks the *Permissions* file and provides information on the permissions you have set up.

Basic Networking Utilities Error Messages

This section lists the error messages associated with the Basic Networking Utilities. There are two types of error messages. ASSERT errors are recorded in the */var/uucp/.Admin/errors* file. STATUS errors are recorded in individual machine files found in the */var/uucp/.Status* directory.

BNU ASSERT Error Messages

When a process is aborted, ASSERT error messages are recorded in */var/uucp/.Admin/errors*. These messages include the file name, SCCS ID, line number, and text listed below. In most cases, these errors are the result of file system problems. The *errno* (when present) should be used to investigate the problem. If *errno* is present in a message, it is shown as ( ) in the following list:

CAN'T OPEN
An open ( ) or fopen( ) failed.

CAN'T WRITE
A write( ), fwrite( ), fprint( ), etc., failed.

CAN'T READ
A read( ), fgets( ), etc., failed.

CAN'T CREATE
A create( ) call failed.

CAN'T ALLOCATE
A dynamic allocation failed.

CAN'T LOCK
An attempt to make a LCK (lock) file failed. In some cases, this is a fatal error.

CAN'T STAT
A stat( ) call failed.

CAN'T CHMOD
A chmod( ) call failed.

CAN'T LINK
A link( ) call failed.

CAN'T CHDIR
A chdir( ) call failed.

CAN'T UNLINK
An unlink( ) call failed.

WRONG ROLE
This is an internal logic problem.

CAN'T MOVE TO CORRUPTDIR
An attempt to move some bad C. or X. files to the */var/uucp/.Corrupt* directory failed. The directory is probably missing or has wrong modes or owner.

CAN'T CLOSE
A close( ) or fclose( ) call failed.

FILE EXISTS
The creation of a C. or D. file is attempted, but the file exists. This occurs when there is a problem with the sequence file access. Usually indicates a software error.

No uucp server
A TCP/IP call is attempted, but there is no server for UUCP.

BAD UID

The uid cannot be found in the */etc/passwd* file. The file system is in trouble, or the */etc/passwd* file is inconsistent.

BAD LOGIN_UID

Same as previous.

ULIMIT TOO SMALL

The ulimit for the current user process is too small. File transfers may fail, so transfer is not attempted.

BAD LINE

There is a bad line in the *Devices* file; there are not enough arguments on one or more lines.

FSTAT FAILED IN EWRDATA

There is something wrong with the ethernet media.

SYSLST OVERFLOW

An internal table in gename.c overflowed. A big/strange request was attempted.

TOO MANY SAVED C FILES

Same as previous.

ENTER FROM fixline ioctl

An ioctl, which should never fail, failed. There is a system driver problem.

BAD SPEED

A bad line speed appears in the *Devices/Systems* files (Class field).

PERMISSIONS file: BAD OPTION

There is a bad line or option in the *Permissions* file. Fix it immediately!

PKCGET READ

The remote machine probably hung up. No action need be taken.

PKXSTART
The remote machine aborted in a non–recoverable way. This can generally be ignored.

IN SEND/SLAVE MODE (INPUT FAILURE)
There was a disconnection in the physical line during file transfer.

SYSTAT OPEN FAIL
There is a problem with the modes of */usr/lib/uucp/.Status*, or there is a file with bad modes in the directory.

TOO MANY LOCKS
There is an internal problem.

XMV ERROR
There is a problem with some file or directory. It is likely the spool directory, since the modes of the destinations were supposed to be checked before this process was attempted.

CAN'T FORK
An attempt to fork and exec failed. The current job should not be lost, but will be attempted later (uuxqt). No action need be taken.

BNU STATUS Error Messages

Status error messages are messages that are stored in the */var/uucp/.Status* directory. This directory contains a separate file for each remote machine that your system attempts to communicate with. These individual machine files contain status information on the attempted communication, whether it was successful or not. What follows is a list of the most common error messages that may appear in these files.

OK
Things are OK.

NO DEVICES AVAILABLE
: There is currently no device available for the call. Check to see that there is a valid device in the *Devices* file for the particular system. Check the *Systems* file for the device to be used to call the system.

WRONG TIME TO CALL
: A call was placed to the system at a time other than what is specified in the *Systems* file.

TALKING
: Self explanatory.

LOGIN FAILED
: The login for the given machine failed. It could be a wrong login/password, wrong number, a very slow machine, or failure in getting through the Dialer–Token–Pairs script.

CONVERSATION FAILED
: The conversation failed after successful startup. This usually means that one side went down, the program aborted, or the line (link) was dropped.

DIAL FAILED
: The remote machine never answered. It could be a bad dialer or the wrong phone number.

BAD LOGIN/MACHINE COMBINATION
: The machine called us with a login/machine name that does not agree with the *Permissions* file. This could be an attempt to masquerade!

DEVICE LOCKED
: The calling device to be used is currently locked and in use by another process.

ASSERT ERROR
: An ASSERT error occurred. Check the */var/uucp/.Admin/errors* file for the error message and refer to the section "BNU ASSERT Error Messages".

SYSTEM NOT IN Systems
The system is not in the *Systems* file.

CAN'T ACCESS DEVICE
The device tried does not exist or the modes are wrong. Check the appropriate entries in the *Systems* and *Devices* files.

DEVICE FAILED
The open of the device failed.

WRONG MACHINE NAME
The called machine is reporting a different name than expected.

CALLBACK REQUIRED
The called machine requires that it calls your system.

REMOTE HAS A LCK FILE FOR ME
The remote site has a LCK file for your system. They could be trying to call your machine. If they have an older version of Basic Networking, the process that was talking to your machine may have failed leaving the *LCK* file. If they have the new version of Basic Networking, and they are not communicating with your system, then the process that has a *LCK* file is hung.

REMOTE DOES NOT KNOW ME
The remote machine does not have the node name of your system in its *Systems* file.

REMOTE REJECT AFTER LOGIN
The login used by your system to login does not agree with what the remote machine was expecting.

REMOTE REJECT, UNKNOWN MESSAGE
The remote machine rejected the communication with your system for an unknown reason. The remote machine may not be running a standard version of Basic Networking.

STARTUP FAILED

Login succeeded, but initial handshake failed.

CALLER SCRIPT FAILED

This is usually the same as "DIAL FAILED." However, if it occurs often, suspect the caller script in the *dialers* file. Use uutry to check.

Glossary

Active System
A system with Basic Networking Utilities and the hardware required to establish communication links (i.e. Auto–Dial Modem).

Address
A number, label, or name that indicates the location of information in the computer's memory.

A.out
The default name of a freshly compiled object file. Historically a.out signifies assembler output.

Archive
(1) A collection of data gathered from several files into one file; (2) especially, such a collection gathered by **ar**(1) for use as a library.

Automatic calling unit
A hardware device used to dial stored telephone numbers. It allows the system to contact another system over phone lines without manual intervention.

Bad block
A section of a storage medium that cannot store data reliably.

Basic Networking Utilities (BNU)
A group of programs and files that permit copy capability between UNIX Systems. Sometimes called UUCP.

Block
The basic unit of buffering in the kernel. See indirect, logical, and physical blocks.

Block device
A device upon which a file system can be mounted; typically a permanent storage device such as a tape or disk drive, so called because data transfers to the device occur by blocks.

Boot
To start the operating system, so called because the kernel must bootstrap itself from secondary storage into an empty machine. No login or process persists across a boot.

Boot block
The first block of a file system that is reserved for a booting program.

Boot program
Loads the operating system into memory.

Buffer
(1) A staging area for input/output where arbitrary–length transactions are collected into convenient units for system operations. (2) To use buffers.

Buffer pool
A region of storage available to the file system for holding blocks. All input/output for block devices goes through the buffer pool so read and write operations may be independent of device blocks.

Cartridge tape
A storage medium that consists of a magnetic tape wound on spools housed in a plastic container.

Character device
A device upon which a file system cannot be mounted such as a terminal or the null device.

Child process
See **fork**.

Controller
A device that directs the transmission of data over the data links of a network.

Crash
If a hardware or software error condition develops that the system can not handle, it takes itself out of service, or crashes. Such conditions occur when the system can not allocate resources, manage processes, respond to requests for system functions, or when the electrical power is unstable.

Cron
A command which creates a daemon that invokes commands at specified dates and times.

Daemon
A background process, often perpetual, that performs a system–wide public function, e.g. **calendar**(1) and **cron**(8).

Device
(1) A special file such as a tape drive or the nulldevice; not a regular file or directory. (2) A physical input/output unit.

Directory hierarchy
The tree of all directories, in which each is reachable from the root via a chain of subdirectories.

Disk
A platter coated with magnetic material on which data can be stored.

Diskette, flex disk
A magnetic storage medium that is smaller and originally more flexible than a hard disk.

Drive
The hardware device that holds magnetic disks, diskettes, and tapes while they are in use.

Dump
A copy of the memory of the system, used for analyzing problems.

Executable file
(1) An object file that is ready to be copied into the address space of a process to run as the code of that process. (2) A file that has execute permission, either an executable file or a shell script.

File system
(1) A collection of files that can be mounted on a block special file; each file of a file system is accessible via some path from the root directory of the file system. (2) The collection of all files on a computer.

Filter
A program that reads from the standard input and writes on the standard output, so called because it can be used as a data–transformer in a pipeline.

Flush
To empty a buffer.

Fork
To split one process into two (the parent process and child process) with separate, but initially identical, text, data, and stack segments.

Free list
In a file system, the list of blocks that are not occupied by data.

Getty
One of a series of processes which may connect the user to the UNIX system. In NCR UNIX System V Release 4, the **ttymon** (and **sac**) processes are normally used instead of **getty**. If used, **getty** is invoked by **init**, and in turn invokes login.

Group
(1) A set of permissions alternative to owner permissions for access to a file. (2) A set of user IDs that may assume the privileges of a group. (3) The group ID of a file.

Group ID
An integer value, usually associated with one or more login names; as the user ID of a process becomes the owner of files created by the process, so the group ID of a process becomes the group of such files.

Hole
A gap in a file caused by seeking while writing. **read**(2) takes data in holes to be zero. A block in a hole occupies no space in its file system. Files with holes in them are known as sparse files.

I–list
The index to a file system listing all the inodes of the file system.

Indirect blocks
Data blocks that are not directly referenced by an inode. The inode has addresses that indirectly reference (by a cascade of pointers) an additional number of blocks to handle extremely large file sizes.

Init
A general process spawner that is invoked as the last step in the boot procedure; it regularly checks a table that defines what processes should run at what run level.

Inode
An element of a file system; an inode specifies all properties of a particular file and locates the file's contents, if any.

Inode number, I–number
The position of an inode in the I–list of a file system.

Interface programs

Shell scripts (for example, those furnished with the LP Spooler software) that interface between the user and some part of the overall system (for example, the printer).

Kernel

The UNIX system proper; resident code that implements the system calls.

Kernel address space

A portion of memory used for data and memory addressable only by the kernel.

Link

(1) To add an entry for an existing file to a directory; converse of unlink. (2) By extension, a directory entry.

Link count

The number of directory entries that pertain to an inode; a file ceases to exist when its link count becomes zero and it is not open.

Load device

Designates the physical device from which a program will be loaded into main memory.

Local system

Refers to the system on the near end of a communication link; normally, your system.

Log files

Contain records of transactions that occur on the system; software that spools, for example, generates various log files.

Logical block

A unit of data as it is handled by the software. The system handles data in logical blocks whose size is determined by the file system.

Memory Image
A copy of of all the segments of a running or terminated program. The copy may exist in main storage, in the swap area, or in a file.

Mode, file mode
The permissions of a file; colloquially referred to by a 3–digit octal number, e.g. 'a 755 file'. See **chmod**(1).

Mount
To extend the directory hierarchy by associating the root of a file system with a directory entry in an already mounted file system. The converse is unmount, spelled **umount**.

Namelist
Same as symbol table.

Network
The hardware and software that constitute the interconnections between computer systems, permitting electronic communication between the systems and associated peripherals.

Networking
For computer systems, this means sending data from one system to another over some communications medium (coaxial cable, phone lines, etc.). Common networking services include file transfer, remote login, remote execution.

Node
A terminating point (system) on a network.

Node name
The name for the system(may be up to 255 characters); used as the official name of the machine in a network. The node name resides in the NODE parameter.

Null device
A device that always yields end–of–file on reading and discards all data on writing.

Object file
A file of machine language code and data; object files are produced from source programs by compilers and from other object files and libraries by the link editor. An object file that is ready to run is an executable file.

Operating system
The program for managing the resources of the computer. It takes care of such things as input/output procedures, process scheduling, the file system, removing this burden from user programs.

Open file
(1) The destination for input or output obtained by opening a file or creating a pipe; a file descriptor. Open files are shared across forks and persist across executes. (2) Loosely, a file that has been opened. However an open file need not exist in a file system, and a file may be the destination of several open files simultaneously.

Other
A set of permissions regulating access to a file by processes with userID different from the owner and groupID different from the group of the file.

Owner
The userid of the process that created a file; the owner has distinctive permissions for a file.

Page
A fixed length, a 4096–byte portion of memory that has a virtual address, and that can be transferred between main and secondary storage.

Paging
The process by which programs are truncated into pages and transferred between main and secondary storage by the virtual handler (or paging daemon).

Parent process
See **fork**.

Partitions
Units of storage space on disk.

Passive system
A system that has Basic Networking Utilities but does not have the hardware required to establish communication links and so never initiates calls.

Permission
A right to access a file in a particular way including read, write, execute (or look up in, if a directory). Permissions are granted separately to owner, group, and others.

Permission bit
A permission, so called because each permission is encoded into one bit in an inode.

Physical block
A unit of data as it is actually stored and manipulated. The system handles data in 512–byte physical blocks.

Physical memory
See **memory**.

Pipe
A direct stream connection between processes, whereby data written on an open file in one process becomes available for reading in another.

Pipeline
A sequence of programs connected by pipes.

Polling
The interrogation of devices by the operating system to avoid contention, determine operation status, or ascertain readiness to send or receive data.

Ports
The point of physical connection between a peripheral device (such as a terminal or a printer) and the device controller, that is part of the computer hardware.

Process
A connected sequence of computation. A process is characterized by a core image with instruction location counter,current directory, a set of open files, control terminal, userid, and groupid.

Process id
An integer that identifies a process.

Process number
Same as process ID.

Profile
(1) An optional shell script, .profile, conventionally used by the shell upon logging in to establish the environment and other working conditions customary to a particular user. (2) To collect a histogram of values of the instruction location counter of a process.

Program
(1) An executable file. (2) A process. (3) All the usual meanings.

Queue
A line or list formed by items in a system waiting for service.

Raw device
A block device, read and write operations to which are not buffered, and are synchronized to natural records of the physical device.

Reboot
Same as boot.

Remote system
Refers to a system on the far end of a communication link; normally, a system that your system calls.

Retension

The process of rewinding the tape in a cartridge tape device to make sure it is at the correct tautness for accurate recording of data.

Root

(1) A distinguished directory that constitutes the origin of the directory hierarchy in a file system. (2) Specifically, the origin for the file system, with the conventional path name "/".
(3) The origin of the directory hierarchy in a file system.

Run level

A software configuration of the system which allows a particular group of processes to exist.

Schedule

To assign resources – main store and CPU time – to processes.

Scheduler

A permanent process, and associated kernel facilities that does swapping.

Search path

In the shell, a list of path names of directories that determines the meaning of a command; the command name is prefixed with members of the search path in turn until a path name of an executable file results; the search path is given by the shell variable PATH.

Section, sector

A 512–byte portion of a track which can be accessed by the magnetic disk heads in the course of a predetermined rotational displacement of the storage device.

Segment
A contiguous range of the address space of a process with consistent storage access capabilities; the four segments are (1) the text segment, occupied by executable code, (2) the data segment, occupied by static data that is specifically initialized, (3) the bss.segment, occupied by static data that is initialed by default to zero values, and (4) the stack segment, occupied by automatic data, see stack. Sometimes (2), (3), and (4) are collectively called data segments.

Semaphore
An IPC facility that allows two or more processes to be synchronized.

Set userId
A special permission for an executable file that causes a process executing it to have the access rights of the owner of the file. The owner's user ID becomes the effective user ID of the process, distinguished from the real user ID under which the process began.

Set userId bit
The associated permission bit.

Shared memory
An IPC facility that permits two or more processes to share the same data space.

Shell
(1) The program **sh(1)**, that causes other programs to be executed on command; the shell is usually started on a user's behalf when the user logs in. (2) By analogy, any program started upon logging in.

Shell script
An executable file of commands taken as input to the shell.

Single user
A state of the operating system in which only one user is supported.

Source file
(1) The uncompiled version of a program. (2) Generally, the unprocessed version of a file.

Special file
An inode that designates a device, further categorized as either (1) a block special file describing a block device, or (2) a character special file describing a character device.

Spool
To collect and serialize output from multiple processes competing for a single output service.

Spooler
A **daemon** that spools.

Startup
Same as boot.

Super block
The second block in a file system, that describes the allocation of space in the file system.

Superuser
userid 0 with access to any file regardless of permissions. The superuser can perform certain privileged system calls, e.g. setting the clock.

Swap
To move the memory image of an executing program between main and secondary storage to make room for other processes.

Swap area
The part of secondary storage to which memory images are swapped. The swap area is disjointed from the file system.

Symbolic link
A file that contains the path name of another file or directory. References to the symbolic link become references to the named file or directory.

Symbol table
Information in an object file about the names of data and functions in that file; the symbol table and address relocation information are used by the link editor to compile object files and by debuggers.

System calls
(1) The set of system primitive functions through which all system operations are allocated, initiated, monitored, manipulated, and terminated. (2) The system primitives invoked by user processes for system–dependent functions, such as I/O, process creation, etc.

System console
The directly connected terminal used for communication between the operator and the computer.

System name
The name (up to 255 characters) for the system; resides in the SYS parameter. This is the name given by the manufacturer, indicating the version of your operating system. It is NOT the node name, a name that uniquely identifies your system for communication with other systems.

Text file, ASCII file
A file, the bytes of which are understood to be in ASCII code.

Track
An addressable ring of sections on a disk or flex disk; each disk or flex has a predefined number of concentric tracks, which allows the disk head to properly access sections of data.

Tunable parameters

Variables used to set the sizes and thresholds of the various control structures of the operating system.

Tuning

(1) Modifying the tunable parameters to improve system performance. (2) The reconfiguration of the operating system to incorporate the modifications into an executable version of the system.

UUCP

A group of programs and files that permit UNIX–to–UNIX copy capability between UNIX Systems. If shown in the text with bold type(**uucp**), this is referring specifically to the **uucp**(1) program or login ID. Also, called Basic Networking Utilities (BNU).

UserID

An integer value, usually associated with a login name. The userID of a process becomes the owner of files created by the process and descendent (forked) processes.

Utility, utility program

A standard, generally useful, permanently available program.

Index

A

E

F

G

H

I

N

O

P

R

S

U

V

W

X