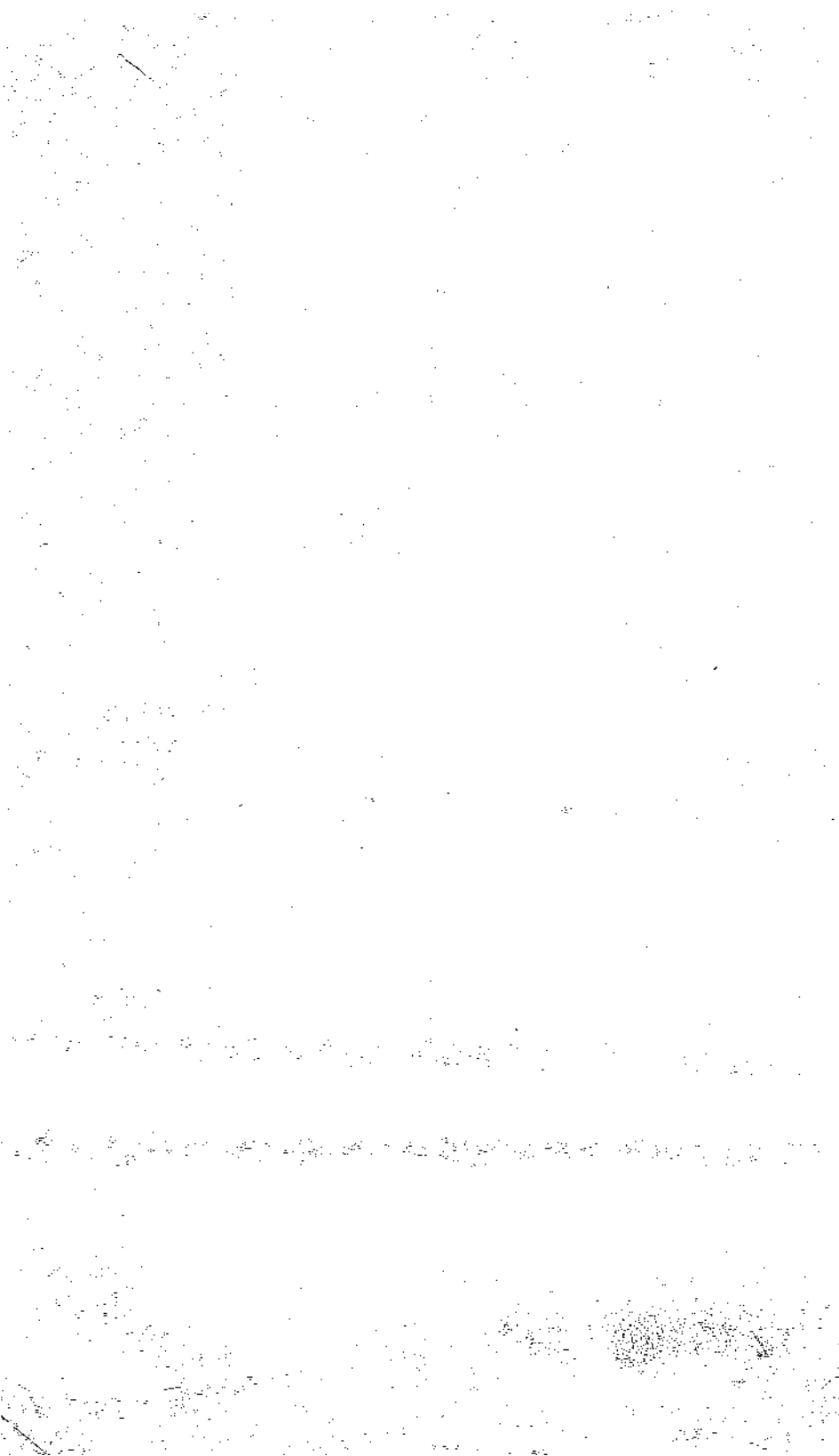


XENIX[®] System V

Operating System

System Administrator's Guide



Information in this document is subject to change without notice and does not represent a commitment on the part of The Santa Cruz Operation, Inc. nor Microsoft Corporation. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy this software on magnetic tape, disk, or any other medium for any purpose other than the purchaser's personal use.

Portions © 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988 Microsoft Corporation.

All rights reserved.

Portions © 1983, 1984, 1985, 1986, 1987, 1988 The Santa Cruz Operation, Inc.

All rights reserved.

ALL USE, DUPLICATION, OR DISCLOSURE WHATSOEVER BY THE GOVERNMENT SHALL BE EXPRESSLY SUBJECT TO RESTRICTIONS AS SET FORTH IN SUBDIVISION (b) (3) (ii) FOR RESTRICTED RIGHTS IN COMPUTER SOFTWARE AND SUBDIVISION (b) (2) FOR LIMITED RIGHTS IN TECHNICAL DATA, BOTH AS SET FORTH IN FAR 52.227-7013.

Hayes is a registered trademark of Hayes Microcomputer Products, Inc.
Microsoft, MS-DOS, and XENIX are trademarks of Microsoft Corporation.



Contents

1 Introduction

- 1.1 Overview 1-1
- 1.2 The System Administrator 1-1
- 1.3 Making Administration Easier with the sysadm Shell 1-2
- 1.4 The Super User Account 1-3
- 1.5 The Keyboard 1-4
- 1.6 About This Guide 1-5

2 sysadmsh: Using the System Administration Shell

- 2.1 Introduction 2-1
- 2.2 Starting sysadmsh 2-2
- 2.3 How the Screen is Organized 2-3
- 2.4 Selecting Menu Items 2-4
- 2.5 Using Forms 2-6
- 2.6 Using Scan Windows 2-11
- 2.7 Getting Help 2-13
- 2.8 Changing the Current Directory within sysadmsh 2-15
- 2.9 The Function Keys 2-15
- 2.10 Using Shell Escapes to Access the XENIX Command Line 2-16
- 2.11 XENIX Commands and sysadmsh Equivalents 2-17

3 Starting and Stopping the System

- 3.1 Introduction 3-1
- 3.2 Starting the System 3-1
- 3.3 Logging In As the Super User 3-3
- 3.4 Stopping the System 3-4

4 Using Filesystems

- 4.1 Introduction 4-1
- 4.2 What is a Filesystem? 4-1
- 4.3 Adding a Second Hard Disk 4-2
- 4.4 Maintaining Free Space In Filesystems 4-14
- 4.5 Filesystems and Large Directories 4-21
- 4.6 Changing/Adding Filesystems on the Primary Hard Disk 4-21
- 4.7 Filesystem Integrity 4-21

5 Maintaining System Security

- 5.1 Introduction 5-1

- 5.2 General Security Practices 5-1
- 5.3 Permissions 5-4
- 5.4 Managing File Ownership 5-8
- 5.5 Changing a User's Password 5-9
- 5.6 Forcing a New Password 5-10
- 5.7 Adding Dial-in Password Protection 5-12
- 5.8 Permitting Users to Mount Filesystems 5-13
- 5.9 Using XENIX Accounting Features 5-14

6 Backing Up Filesystems

- 6.1 Introduction 6-1
- 6.2 Strategies for Backups Using sysadmin 6-1
- 6.3 Preparations for Scheduled Backups 6-3
- 6.4 Performing a Scheduled Backup 6-9
- 6.5 Performing an Unscheduled Backup 6-13
- 6.6 Getting a Backup Listing 6-16
- 6.7 Restoring Individual Files or Directories From Backups 6-17
- 6.8 Restoring an Entire Filesystem 6-20
- 6.9 Editing /etc/default/filesys and /etc/default/archive 6-25
- 6.10 An Explanation of Backup Levels 6-27

7 Adding Device Drivers with the Link Kit

- 7.1 Introduction 7-1
- 7.2 Device Drivers 7-1
- 7.3 Freeing Kernel Space for Drivers 7-9
- 7.4 Testing and Installing the New Kernel 7-10

8 Tuning System Performance

- 8.1 Introduction 8-1
- 8.2 Reallocating Kernel Resources with configure 8-2
- 8.3 Reconfiguring Because of Persistent Error Messages 8-4
- 8.4 Reconfiguring for Performance 8-5
- 8.5 Defining Efficient System Usage Patterns 8-9
- 8.6 Using vmstat to Diagnose System Inefficiency 8-11
- 8.7 Summary of Tunable Parameters 8-14

9 Using DOS and XENIX On the Same Disk

- 9.1 Introduction 9-1
- 9.2 Partitioning the Hard Disk Using fdisk 9-1
- 9.3 Installing XENIX on a DOS System 9-4
- 9.4 Using XENIX and DOS With Two Hard Disks 9-6
- 9.5 Removing an Operating System From the Hard Disk 9-7
- 9.6 DOS Accessing Utilities 9-7
- 9.7 XENIX and DOS On Non-Standard Disks 9-8

10 Preparing XENIX for Users

- 10.1 Introduction 10-1
- 10.2 Adding a User Account 10-1
- 10.3 Creating a Group 10-6
- 10.4 Changing a User's Login Group 10-7
- 10.5 Changing a User ID 10-9
- 10.6 Removing a User Account 10-11
- 10.7 Changing XENIX Initialization 10-13

11 Building a Remote Network with UUCP

- 11.1 Introduction 11-1
- 11.2 Connecting Two Local Systems Using A Direct Wire 11-9
- 11.3 Connecting Remote UUCP Systems with a Modem 11-11
- 11.4 Configuring UUCP on Your System 11-17
- 11.5 Special UUCP Configuration Options 11-44
- 11.6 Administrating Your UUCP System 11-50
- 11.7 Troubleshooting 11-57
- 11.8 Keeping Traffic and Congestion Under Control 11-59
- 11.9 UUCP Error Messages 11-61

12 Building a Local Network With Micnet

- 12.1 Introduction 12-1
- 12.2 Planning a Network 12-1
- 12.3 Building a Network 12-8
- 12.4 Starting the Network 12-16
- 12.5 Testing a Micnet Network 12-17
- 12.6 Using a UUCP System 12-20

13 XENIX Directories and Special Device Files

- 13.1 Introduction 13-1
- 13.2 XENIX Directories 13-1
- 13.3 Log Files 13-6
- 13.4 Special Device Files 13-7

14 Using Terminals and Modems

- 14.1 Introduction 14-1
- 14.2 Using MultiscreenTM 14-1
- 14.3 Adding and Configuring Serial Ports 14-2
- 14.4 Setting Up a Serial Console 14-4
- 14.5 Adding a Terminal 14-4
- 14.6 Setting Terminal Lines 14-8
- 14.7 Changing Serial Line Operation 14-13

- 14.8 Setting the Terminal Type 14-15
- 14.9 Setting the Terminal Type Automatically 14-16
- 14.10 Removing a Terminal 14-17
- 14.11 Modem Usage under XENIX 14-18

15 Using Printers

- 15.1 Introduction 15-1
- 15.2 The Printer Spooling System 15-1
- 15.3 Installing a Printer 15-2
- 15.4 Stopping the Print Spooling Daemon: lpsched 15-8
- 15.5 Creating an Init Device File 15-9
- 15.6 Moving Requests Between Printers: lpmove 15-10
- 15.7 Controlling Print Requests: accept 15-11
- 15.8 Canceling a Print Request 15-12
- 15.9 Enabling and Disabling Printers 15-12
- 15.10 Printer Interface Programs 15-13
- 15.11 Adding a Local Printer 15-15

16 Using Floppy Disks and Tape Drives

- 16.1 Introduction 16-1
- 16.2 Using a Cartridge Tape Drive 16-1
- 16.3 Using Floppy Disks 16-6

17 Using Bus Cards

- 17.1 Introduction 17-1
- 17.2 Installing Bus Cards 17-1
- 17.3 Adding Additional Memory 17-3

18 Using a Mouse With XENIX

- 18.1 Introduction 18-1
- 18.2 Installing the Hardware 18-1
- 18.3 Installing A Mouse 18-1
- 18.4 Using the Mouse 18-6

19 Solving System Problems

- 19.1 Introduction 19-1
- 19.2 Restoring a Nonechoing Terminal 19-1
- 19.3 Restarting a Locked Terminal 19-1
- 19.4 Loading adb(CP) Without the Development System 19-4
- 19.5 Fixing Console Keyboard Lock-up 19-6
- 19.6 Restarting a Stopped Printer Queue 19-8
- 19.7 Fixing a Slow Parallel Printer 19-9
- 19.8 Stopping a Runaway Process 19-11

- 19.9 Replacing a Forgotten User Password 19-12
- 19.10 Removing Hidden Files 19-12
- 19.11 Restoring Free Space 19-13
- 19.12 Restoring Lost System Files 19-13
- 19.13 Restoring a Corrupted root Filesystem 19-13
- 19.14 Recovering from a System Crash 19-14
- 19.15 Fixing Bad HZ Value 19-15
- 19.16 Recovering from a General Protection Trap 19-16
- 19.17 Mapping a Bad Track 19-17



Chapter 1

Introduction

- 1.1 Overview 1-1
- 1.2 The System Administrator 1-1
- 1.3 Making Administration Easier with the sysadm Shell 1-2
- 1.4 The Super User Account 1-3
- 1.5 The Keyboard 1-4
- 1.6 About This Guide 1-5



1.1 Overview

The XENIX operating system is a powerful system of programs that allows you to accomplish a full spectrum of tasks, from developing high-level and assembly language programs to creating, editing, and typesetting documents. To keep running smoothly, the powerful XENIX system requires careful control of its operation and a regular schedule of maintenance. This guide explains how to operate and maintain the XENIX operating system on your computer, ensuring maximum performance with the fewest system problems.



This guide also explains how to expand a XENIX system with remote and local networks. For local networking over serial lines, **micnet** can link XENIX systems in your work environment. For remote communications over phone lines, UUCP can be set up to communicate with XENIX/UNIX sites all over the world. (See “Building a Local Network with Micnet,” and “Building a Remote Network with UUCP” in this Guide for a complete explanation of network facilities available.)

1.2 The System Administrator

Every XENIX system should have one person in charge of system maintenance and operation. In this guide, that person is called the system administrator. It is the system administrator's duty to ensure the smooth operation of the system and to perform tasks that require special privileges. These duties require that the system administrator become proficient with a wide variety of XENIX functions.

Depending on the size of the system and the number of users on it, a system administrator's job can be anything from a once-a-day task to a full-time job. Even if the system is small, the system administrator should faithfully perform each required maintenance task, since sloppy maintenance can adversely affect XENIX performance.

The system administrator should keep a log of all system modifications and system events. Each event, message, backup, or modification should be logged with the date, time, and name of the person logging, and the circumstances surrounding the event. For example, if a new application is added to the system software, an entry should be placed in the log. This entry should include the time, date, and name of the person installing, and any notes about the software or installation that may be helpful. An accurate log helps in diagnosing system problems and charting the growth and use of a system.

All tasks in this guide are presented from the system administrator's point of view, but many can also be accomplished by ordinary users. Since

1 some of the tasks dramatically change the system's operation, we recommend that, whenever possible, the system administrator perform these tasks. However, no matter who performs an operation, it should be logged in the system log. Following these rules can prevent unwanted or unnecessary changes to the system.

The System Administrator has several tasks to perform, sometimes on a daily basis:

- Making certain that adequate backups (regular copies of files on the system) are made and stored for future use.
- Handling problems related to use of limited computer resources (disk space, number of processes, etc.)
- Alleviating system communication (network) stoppages due to failed connections.
- Applying operating system updates and maintenance fixes.
- Providing general support to users.

1.3 Making Administration Easier with the `sysadm` Shell

The `sysadmsh` is a menu interface designed to simplify the task of system administration. The menus, submenus and screens allow you to simply point and pick, or fill in blanks. The `sysadmsh` allows less-experienced system administrators to use XENIX commands that would otherwise require memorization and constant referring to manual pages. The `sysadmsh` includes context-sensitive help; simply press the key from any menu to display further explanations of the menu options.

If you are new to XENIX, we strongly recommend that you become familiar with the concepts and tasks covered in the *XENIX Tutorial*, a tutorial for new users. This guide assumes some familiarity with XENIX; after studying the *XENIX Tutorial*, you should be able to perform the basic system administrative tasks described here.

To aid users of the `sysadm` shell, the documentation of this guide is supplemented by `sysadmsh` references that appear below XENIX command line instructions.

For example, the following instructions refer to the `custom` utility, used to add more software to your system. Below the actual command is a sequence of `sysadmsh` menu selections.

Enter the following command:

custom

Δ **sysadmsh** users select: System→Add→Software



This means that you can access the functions of the **custom** command by first selecting “System” at the main **sysadmsh** menu, followed by selecting “Addition” at the next lower level, and finally “Software” at the lowest level. Selections can be made from the menu in any of the following ways:

- Tab through the menu options using the TAB key and press RETURN on the option you want.
- Move left and right through the options using the arrow keys and press RETURN on the desired option.
- Press the first letter of the option desired. This is the quickest way. Using the example above, you would simply enter “sas” (without the RETURN key) to reach the **custom** menu.

For more instructions on using the **sysadmsh**, refer to the “sysadmsh: Using the System Administration Shell” chapter in this guide.

1.4 The Super User Account

The super user account is a special account for performing system maintenance tasks. It gives the system administrator unusual privileges that ordinary users do not have, such as accessing all files in the system, and executing privileged commands. Many of the tasks presented in this guide require that the system administrator be logged in as the super user. To do this, the system administrator must know the super user password created during the installation of the XENIX system. (See the *XENIX Installation Guide*.)

Log in as the super user only to perform system-maintenance tasks. Even if the system administrator is the only one using the system, that person should create a user account for day-to-day work, reserving the super user account for system-maintenance tasks only.

Few users should know the super user password. Misuse of the super user powers by naive users can result in a loss of data, programs, and even the XENIX system itself.

1.5 The Keyboard



Many keys and key combinations have special meanings in the XENIX system. These have names that are unique to the XENIX system, and may not correspond to the keytop labels on your keyboard. To help you find these keys, the following table shows which keys on a typical terminal correspond to those on the XENIX system. A list for your particular login device is in **keyboard(HW)**.

In this table, a hyphen (-) between keys means “hold down the first key while pressing the second.”

Special Keys

XENIX Name	Keytop	Action
BREAK	Delete	Stops the current program, returning to the shell prompt. This key is also known as the INTERRUPT or DELETE key.
BACKSPACE	Backspace	Deletes the character to the left of the cursor.
Ctrl-d	Ctrl-d	Signals the end of input from the keyboard; exits current shell or initiates the “logout” procedure if the current shell is the login shell.
Ctrl-h	Erase	Deletes the first character to the left of the cursor. Also called the ERASE key.
Ctrl-q	Ctrl-q	Restarts printing after it has been stopped with Ctrl-s.
Ctrl-s	Ctrl-s	Stops printing at the standard output device, such as a terminal. Does not stop the program.

Ctrl-u	Ctrl-u	Deletes all characters on the current line. Also called the KILL key.
Ctrl-\	Ctrl-\	Quits current command and creates a <i>core</i> file. (Recommended for debugging only.) See core(F) for more information.
ESCAPE	Esc	Exits the current mode; for example, exits insert mode when in the editor vi .
RETURN	Return	Terminates a command line and initiates an action from the shell.

Many of these special function keys can be modified by the user. See **stty(C)** for more information.

1.6 About This Guide

The tasks presented in this guide range from simple ones requiring very little knowledge about XENIX, to complex tasks requiring extensive knowledge about XENIX and your computer.

Each chapter explains the tools and knowledge you need to complete the tasks described in that chapter. In some cases, you may be referred to other manuals, such as the *XENIX User's Guide* or the *XENIX User's Reference*.

This guide contains chapters about computer hardware you may wish to use with XENIX. The use and interaction of various devices with XENIX is described in a comprehensive fashion. For example, "Using Floppy Disks and Tape Drives" discusses the use of magnetic storage media, and covers the basics of preparing XENIX for such a device, installing it, and how to use the drive once it has been installed.

In addition, there are chapters dealing with several other types of devices you may wish to use, and there are many chapters to help you administer your system. Some are designed to help you set up a network with other computer systems and some help you maintain and understand your own system.

XENIX System Administrator's Guide

Pay special attention to the chapters “Backing Up Filesystems” and “Solving System Problems.” The latter is an excellent resource to help you keep your system running smoothly, and the chapter on backups discusses the most important aspect of system administration.



Chapter 2

sysadmsh: Using the System

Administration Shell

- 2.1 Introduction 2-1
- 2.2 Starting sysadmsh 2-2
- 2.3 How the Screen is Organized 2-3
- 2.4 Selecting Menu Items 2-4
- 2.5 Using Forms 2-6
- 2.6 Using Scan Windows 2-11
- 2.7 Getting Help 2-13
- 2.8 Changing the Current Directory within sysadmsh 2-15
- 2.9 The Function Keys 2-15
- 2.10 Using Shell Escapes to Access the XENIX Command Line 2-16
- 2.11 XENIX Commands and sysadmsh Equivalents 2-16



2.1 Introduction

The **sysadmsh** (system **administration shell**) is a menu interface designed to simplify the task of system administration. The **sysadmsh** allows you to run the scores of system administration commands with their numerous options without having to use the traditional XENIX command-line.

This chapter explains how to use the **sysadmsh** interface. In order to use **sysadmsh** effectively, you also need to know something about the XENIX commands called by **sysadmsh**. A list of these commands and their corresponding **sysadmsh** menu options is given at the end of this chapter. This list refers you to other sections of the XENIX documentation that contain detailed information on each command.

You will find it easier to learn the material in this chapter if you start the **sysadmsh** and actually run the examples as you get to them. The chapter assumes that you have some knowledge of the XENIX operating system. You should become familiar with the concepts covered in the *XENIX Tutorial* before using the **sysadmsh** menus.



2.2 Starting sysadmsh

There are two ways to start **sysadmsh**:

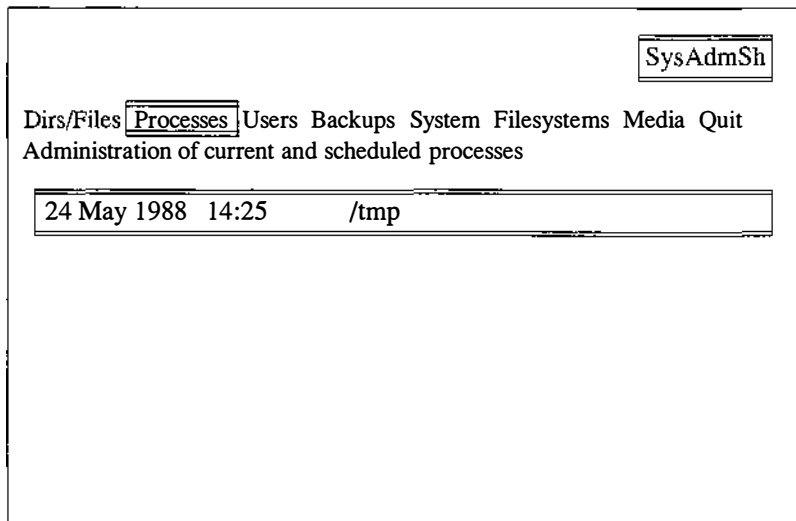
- Log in to the system as user **sysadm**, or
- While already logged in as **root**, enter the following command:

sysadmsh

For the purposes of this tutorial, log in as **root** and enter the following commands:

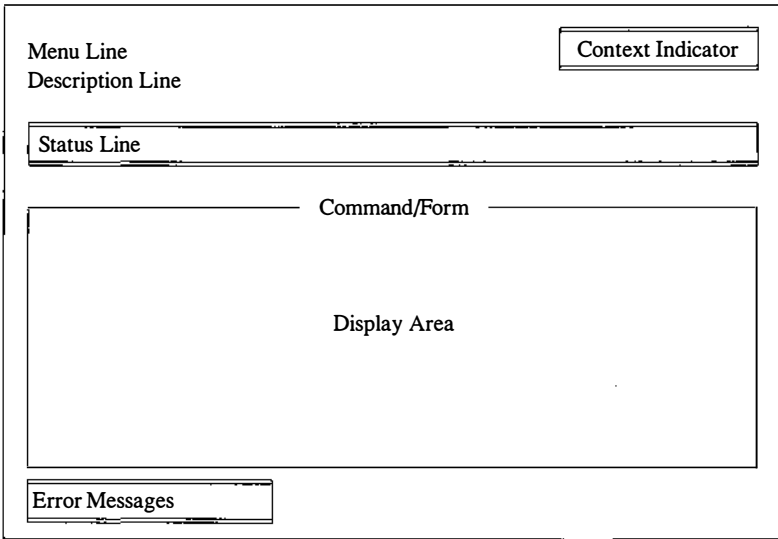
```
cd /tmp
sysadmsh
```

The main **sysadmsh** menu is displayed:




2.3 How the Screen is Organized

A schematic of the **sysadmsh** screen is given below. Areas shown between double lines appear on the screen as highlighted areas or bars of text. Each area is used to display specific types of information:



2

- The Context Indicator is the highlighted bar of text in the upper-right corner of your screen. It displays the name of the current menu. The Context Indicator for the **sysadmsh** opening screen shows *SysAdmSh*.
- The Menu Line displays the menu options that are currently available. The main **sysadmsh** menu consists of eight options: System, Filesystem, Dirs/Files, Backups, Users, Processes, Media, and Quit.
- The Description Line gives you a brief description of the currently highlighted menu option.
- The Status Line is the highlighted bar of text that separates the Menu and Description Lines from the Display Window. The Status Line in the **sysadmsh** opening screen contains the date, time, and current working directory. When a XENIX command is being executed, the name of the command and the options being used are displayed at the far right of the Status Line.

- 
- The Command/Form line displays a title for the contents of the Display Area. This can be either a XENIX command name, or the name of a **sysadmsh** form. When a command name is displayed, the location of the manual page associated with the command is appended in parentheses. For example, when the **who** command is being run, the Command Line displays “who(C).” This means that the command can be found in the (C) Section of the *XENIX Reference*.
 - The Display Area is where **sysadmsh** forms and scan windows are displayed. Forms and scan windows are explained in detail later in this chapter.
 - Error Messages and recovery instructions are displayed on the last line of the screen in highlighted text.

2.4 Selecting Menu Items

The keystrokes listed below are used to traverse the menus. Note that there are several ways to select options; if you have used menu-based programs before, use the method you are most familiar with.

Basic sysadmsh Keystrokes

Action	Keystroke
Move to menu option	Arrow keys, or <SPACEBAR>
Select menu option	First letter of option, or move highlight to option and press <RETURN>
Retreat to previous menu	<ESC>
Get help	<F1>

You can familiarize yourself with the menu options by using the Arrow keys or <SPACEBAR> to move the highlight from option to option. Each time you move the highlight to a new option, a description of that option appears on the Description Line.

sysadmsh has a hierarchical menu structure. Many of the menu options move you down to another menu. For example, when you select the Processes option from the main menu, a menu containing options that let you check on and manipulate your machine's processes is displayed. The menu hierarchy makes it easy to find the command you need by moving down from one menu to the next. Eventually you get to a menu option that either executes a XENIX command or displays a form that you must

fill in with the details that the command needs. Note that typing the first letter of the option name is the quickest way to move through menu levels; in time you will be able to instantly reach the function you need by pressing three- and four-letter codes you have memorized.

The best way to learn how to use menus is to practice making menu selections with these keystrokes. If you select an option by mistake, you can always retreat to the previous menu by pressing the <ESC> key. If you are several levels deep, you can return to the Main menu by pressing the <F2> key and then typing **n**. <F2> takes you to the Quit option, and **n** returns you to the Main menu. To help you find your way through the **sysadmsh** menus, here is a chart of the second level menus:



Menu Map

Dirs/Files	Processes	Users	Backups	System	Filesystems	Media
↓	↓	↓	↓	↓	↓	↓
List	Report	Report	Create	Report	Check	List
View	Terminate	Add	Restore	Halt	Add	Extract
Copy		Delete	Schedule	Execute	Mount	Archive
Edit		Modify		Configure	Umount	Format
Modify		Communicate		Add		Duplicate
Print				Delete		Tapedump
Archive						
Differences						
Remove						
UseDOS						

This chapter uses a syntax convention for denoting a string of menu options. For example, to print a file you must select the Dirs/Files option from the main menu, and then select the Print option from the Dirs/files menu. This sequence is denoted by the shorthand notation Dirs/files→Print, and can be executed by typing **dp**.

When you select a menu option, one of three things happens:

- A lower level menu is displayed,
- You are dropped into a form, or
- A XENIX command is executed and the result displayed in a scan window.

The next two sections explain the details of forms and scan windows.

2.5 Using Forms

Some menu options require additional information in order to perform the correct task. For example, the **Print** option cannot do anything until you tell it what you want to print and which printer to use. When you select an option of this sort, a form appears on the screen. By filling in the form, you give the command the information it needs.

2

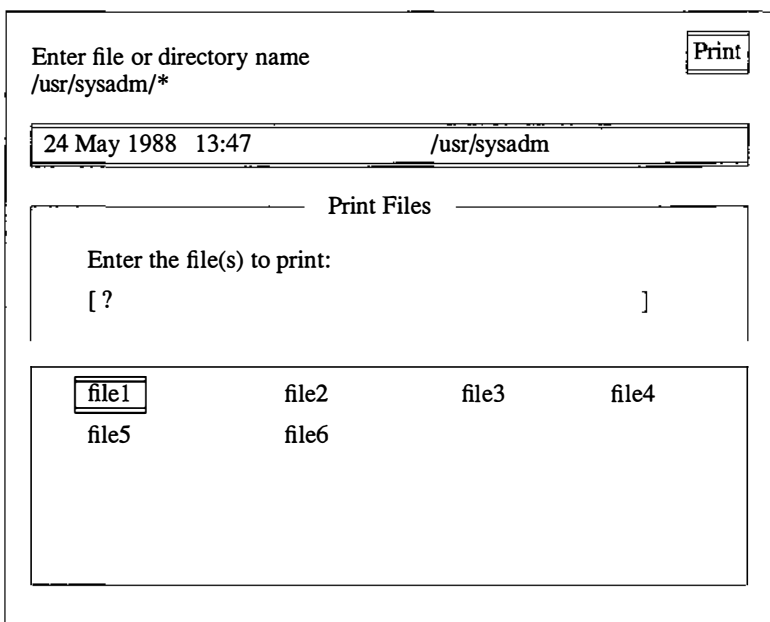
The example below demonstrates how forms work by showing you how to print a file in your current directory. After the example, the keystrokes that allow you to move around the form, edit it, and select "Point and Pick" choices are listed.

To print a file, first select Dirs/Files→Print. The Print menu is displayed:

Enter file or directory name	<input type="button" value="Print"/>								
<table border="1"> <tr> <td>24 May 1988 13:47</td> <td>/usr/sysadm</td> </tr> </table>		24 May 1988 13:47	/usr/sysadm						
24 May 1988 13:47	/usr/sysadm								
Print Files									
<table border="1"> <tr> <td colspan="2">Enter the file(s) to print:</td> </tr> <tr> <td>[<input]<="" td="" type="text" value="?"/> <td>]</td> </td></tr> <tr> <td colspan="2">Enter the name of the printer to send the files to:</td> </tr> <tr> <td>[<input]<="" td="" type="text" value="?"/> <td>]</td> </td></tr> </table>		Enter the file(s) to print:		[<input]<="" td="" type="text" value="?"/> <td>]</td>]	Enter the name of the printer to send the files to:		[<input]<="" td="" type="text" value="?"/> <td>]</td>]
Enter the file(s) to print:									
[<input]<="" td="" type="text" value="?"/> <td>]</td>]								
Enter the name of the printer to send the files to:									
[<input]<="" td="" type="text" value="?"/> <td>]</td>]								

sysadmsh: Using the System Administration Shell

Notice that the highlight is on the first item in the form, and that a question mark (?) appears there. The question mark means that you can obtain a list of choices by pressing <F3>. You can enter the filename if you know it, but for the sake of this tutorial, assume you need to find the filename and press <F3> now. A window opens up overlapping part of the Print form:



The screenshot shows a terminal window with the following content:

```
Enter file or directory name
/usr/sysadm/*
```

```
24 May 1988 13:47 /usr/sysadm
```

Print Files

Enter the file(s) to print:

[?]

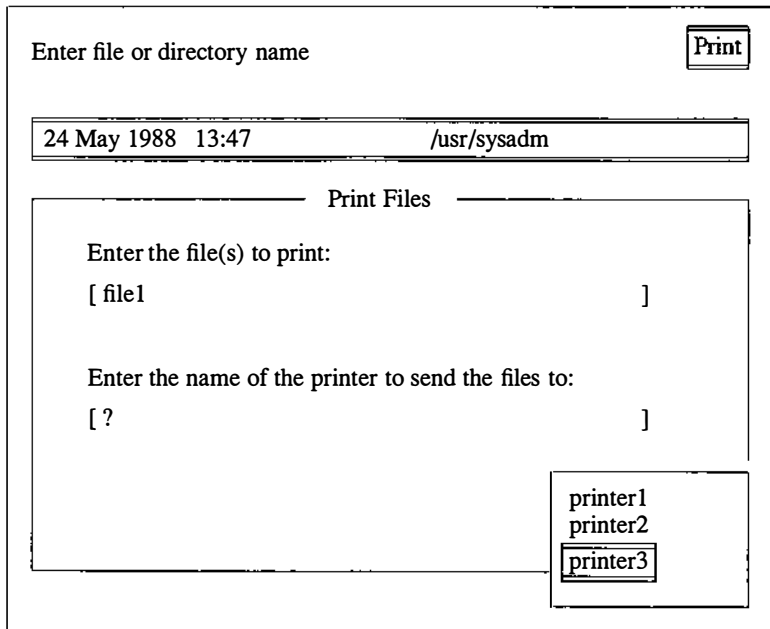
file1	file2	file3	file4
file5	file6		

2

The window contains a list of files that you can select. To select a file, “point” to it by moving the highlight to it, and “pick” it by pressing return. This is known as “Point and Pick,” and is used whenever a range of choices is displayed. When you have made your selection, the window closes and you are returned to the Print form.

Note that the name of the file you selected is now displayed in the form. You can now change the name using the edit keys (listed later in this section), or press <RETURN> to move to the next field.

Note that a question mark appears here also. Now you should enter the name of the printer to be used. If you don't know the printer name, press <F3> and another, smaller window is opened that contains a list of installed printers:



Enter file or directory name Print

24 May 1988 13:47 /usr/sysadm

Print Files

Enter the file(s) to print:
[file1]

Enter the name of the printer to send the files to:
[?]

printer1
printer2
printer3

You can select the printer just as you did the name of the file. When you have selected a printer, you are returned to the Print menu. Press <RETURN> once more to exit the form and send the file to the printer. If you want to exit the form without executing it, press <ESC>.

The keystrokes listed in the following tables allow you to use forms easily:

Movement, Quit, and Execute Keys

Keystroke	Action
<ESC>	Tells the program that you have changed your mind and do not want to finish filling in this form. The form is removed, and no action performed. You are returned to the previous menu.
Up, Down Arrow	Moves to other fields in a form. Some fields are restricted and no input is allowed. The Arrow keys will skip over these. Other fields must be filled in. Pressing the Down Arrow key on the last item in a list brings you back to the first item.
Left, Right Arrow	Moves left and right in the current field. Allows changing of text without retyping the entire line.
<RETURN>	Pressing <RETURN> on a field completes the data entry to that field, and moves the cursor to the next field. In the last field, <RETURN> completes the entire form and tells the shell that the data is ready to use.
<CTL> g	Exits and executes the form from wherever you are. Think <i>g</i> for <i>go</i> .
Other keys	You can use the spelling checker utility when you are in a form. If you think a word might be misspelled, press <F4> while the cursor is on the word and a list of possible correct spellings will appear in a Point and Pick list. The word you select replaces the misspelled word.



Edit Keys

Keystroke	Action
<CTL> x	Delete the current line, start over.
<CTL> w	Delete the current word.
<CTL> a	Move the cursor to the beginning of the line.
<CTL> z	Move the cursor to the end of the line.
<CTL> o	Toggle into or out of overstrike mode.
<BACKSPACE>	Back up and delete one character (left of cursor).
Left, Right Arrow	Move on the edit line.

Point and Pick Keys

Keystroke	Action
<RETURN>	Pressing <RETURN> on a item name selects the item.
<ESC>	No item is desired, abort the selection process. The list is removed and no action performed.
Up, Down Arrow	Move to other items in a list.
Left, Right Arrow	Move across a multi-column display.
<SPACEBAR>	When the application will accept more than one item, the space bar is used to mark them. A marked item is indicated by a “#” character in the left column. It may be unmarked by pressing the space bar a second time while on the item. The entire collection of marked items is selected by pressing <RETURN>.
<F5>	The “Search” key is useful for finding items in long listings. A prompt appears and you enter the string to search for, and press <RETURN>. If the item is found, the highlight moves to that item, and another <RETURN> selects the item. If no match is found the highlight does not move.

First letter The fastest method of selecting an item is by its first letter. Press the first letter of the item and the highlight moves to that item. Pressing <RETURN> then selects the item. If several items begin with the same letter, the cursor will move to the first occurrence in the list.

2.6 Using Scan Windows

When you execute a XENIX command by selecting a **sysadmsh** menu option, the result of the command is displayed in a Scan Window. Scan Windows are also used to display the contents of files and directory listings. To demonstrate the use of Scan Windows, let's say you want to know who is currently logged on to the system. To do this, select Users→Report→Current. (This runs the XENIX **who(C)** command.)

When you select the Current option, a Scan Window that displays the output of the **who(C)** command appears in the Display Area:

Current

<esc> to exit, movement keys are active

24 May 1988 13:47
who -H

who(C)

NAME	LINE	TIME
root	tty01	24 May 10:23
faithz	tty02	24 May 11:03
faithz	tty05	24 May 10:55
karlc	tty07	24 May 07:03
teresae	tty08	24 May 08:21
barta	tty09	24 May 12:52
stevem	tty10	24 May 08:45
mattb	tty14	24 May 09:34

Note that the name of the command (who) and the XENIX reference section in which its description can be found (C) are displayed at the top of the window. Also note that the option given to the command (-H) is displayed in the right hand side of the Status Line. If you do not understand the information displayed, look up the command in the Xenix Reference for more information.

2

You can recognize a Scan Window by the vertical "scroll bar" that appears at the extreme right edge of the window. When the window is at the top of your text, the asterisk at the top of the Scroll Bar is visible. If it is at the bottom, the asterisk at the bottom of the Scroll Bar is visible. You can see both asterisks when the window contains all of the text.

The scroll bar also indicates where you are in the window. The highlighted portion of the bar represents the section of text that is currently displayed in the window. As you scroll up and down, the highlighted bar moves with you.

Use these keys when you are in a scan window:

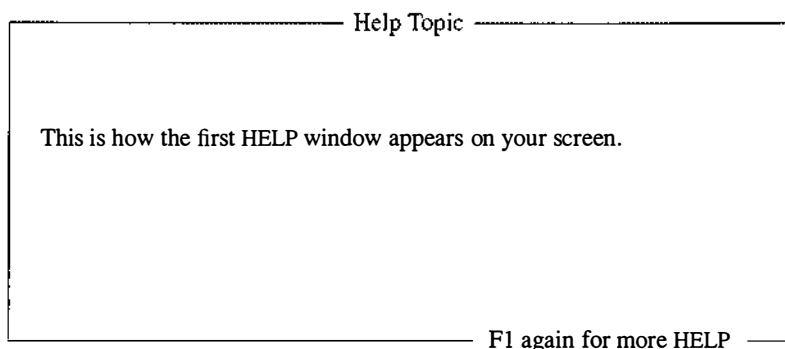
Scan Keys

Action	Keystroke
Exit the file	<ESC>
Move up one line	Up Arrow
Move down one line	Down Arrow, or <RETURN>
Move down a page	PgDn, or <SPACEBAR>
Move up a page	PgUp
Move to top of display	Home
Move to bottom of display	End
Search for a pattern in display	<F5>
Print the output of command or file currently in Scan Window	<F7>

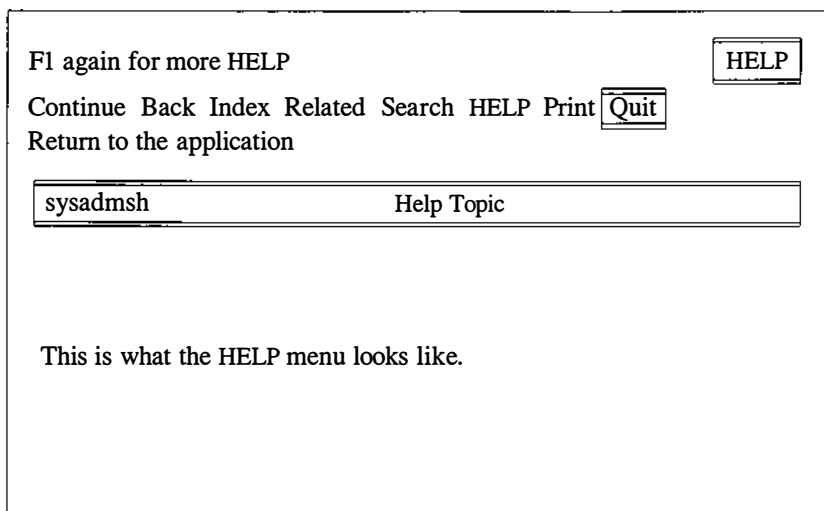
2.7 Getting Help

You can press the <F1> key to display more information to help you with your selection.

When you press the <F1> key, a Help window opens within your current screen. It looks like this:



The window contains some basic information. If you need more help, you can press <F1> again and the complete Help menu is displayed:



From this menu you can select a variety of more detailed information. When you are finished, select Quit from the Help menu and you will be returned to your place in the `sysadmsh` menu proper.

Here are the menu options for Help:

Help Options

Option	Action
Continue	Continue on to the next page. All the vertical movement keys are active: Up and Down Arrows, Page Up and Down, Home and End. If there is no further information, the highlight moves to the Help menu Quit option and the Description Line reads "Return to the application."
Back	Move back to topics that have been seen previously. There is no corresponding "Forward." This is also used to back up to more general topics. You can go back until the top-level introductory topic is reached.
Index	Choose a new topic from a list of indexed topics.
Related	Choose a new topic related to the current one.
Search	Search for a new topic by matching a pattern. First, you specify where to look (the titles, the text lines or both), and then give the pattern. The pattern can be a simple keyword (like "create" or "date") or a more complex "regular expression." A list of topics containing the pattern is presented.
Help	How is the help facility itself used? A table similar to this one is displayed on the screen. If you need further information, look for your topic in Index, Related, or Search.
Print	Make a hardcopy printout of the current topic. First, you select which printer to use, then choose Go from the submenu to send the printout.
Quit	Exit Help and return to <code>sysadmsh</code> . <F2> or <ESC> are other ways to exit quickly.

Each Help screen has general information available, as well as specific information about each option listed on the menu from which Help was selected. Each descriptive passage is preceded by the associated Menu Line and followed by a reference to the XENIX Operating System

documentation. Characters in parentheses following a command name indicate a section of the *XENIX Reference*. For example, **vi(C)** denotes that there is information about **vi** in the *XENIX Reference* (C) section.

Note

When you are within an actual XENIX command, you do not have access to the Help facility. For example, when you select: Dirs/Files→Edit, you are within the XENIX **vi** command, and the **sysadmsh** keys no longer function. When you exit the command and return to the **sysadmsh**, the keys will function as expected. If no element of the **sysadmsh** is visible on the screen (Menu Line, boxes, Context Indicator, etc.) then Help will probably not be available. If you need help, exit from the current process and press the <F1> key to view Help. In general, it is best to use Help prior to executing a menu selection.




2.8 Changing the Current Directory within sysadmsh

There are many occasions when it is necessary to change your current directory to use certain files and commands. You can move to another directory by pressing the <F6> key. The current directory is displayed at the top of the screen. You can use the <BACKSPACE> key to erase the name of the current directory (to start over), or you can add to or alter part of the current name. When you press <RETURN>, your directory change will be executed.

2.9 The Function Keys

The function keys give you access to several time-saving features.

Key	Action
<F1>	Help key - displays help for the current context within the application. Further information is available by pressing <F1> again.

- 
- <F2> Exit key - activates the Quit option on the top menu-level. This is an opportunity to return to the application. For some files, there is a choice of final status at this point.
 - <F3> List key (used within a form) - displays a list of items that are acceptable for the current field. This is similar to entering a “?” in a form entry.
 - <F4> Spell key (used within a form) - displays a list of words that are possible correct spellings of the word in the current field. Select a word from the list by pressing <RETURN>. The word is then placed in the field.
 - <F5> Search key (used within a window) - prompts for a string to search for. When you enter a string and press <RETURN>, the highlight moves to the item in the list that matches the pattern. If no match is found the search fails and the highlight does not move.
 - <F6> New directory - offers the opportunity to change your current working directory. Note that this will not change the directory you were in when you invoked **sysadmsh**.
 - <F7> Print key - prints the output of any command which is displayed in a Scan Window.

2.10 Using Shell Escapes to Access the XENIX Command Line

You can execute a XENIX command by typing the shell-escape character, an exclamation point (!). The menus are replaced by a subshell that displays a text-entry line and a prompt asking for a command. When you enter the command and press <RETURN>, the command is executed by XENIX. After the command is completed, the output is displayed on the screen and you are prompted to press any key to return to the shell.

2.11 XENIX Commands and sysadmsh Equivalents

The following table lists the XENIX commands that the various **sysadmsh** menu options execute. For more information on a particular command, see the manual page for the command. The reference section containing each command's manual page is given in parentheses after the command name. The C, M, and F reference sections are in the XENIX User's Reference; the ADM section is in the XENIX System Administrator's Guide.

sysadmsh option	XENIX command
Dirs/Files →	
List	ls(C) -CF
View	cat(C)
Copy	copy(C)
Edit	edit (SCO Lyrix, vi(C) , ed(C) or definable via environment variable)
Modify→	
Permissions	chmod(C)
Ownership	chown(C)
Group	chgrp(C)
Name	mv(C)
Size	compress(C)
Format	translate(C)
Print	lp(C) (definable via environment variable)
Archive	tar(C) -c
Differences	dircmp(C) or diff(C)
Remove	rm(C) -rf
UseDOS→	
List	dos(C) : <i>dosdir</i> or <i>dosls</i>
Remove	dos(C) : <i>dosrmdir</i> or <i>dosrm</i>
MakeDir	dos(C) : <i>dosmkdir</i>
Copy	dos(C) : <i>doscp</i>
View	dos(C) : <i>doscat</i>
Format	dos(C) : <i>dosformat</i>
Processes →	
Report	ps(C)
Terminate	kill(C)
Users →	
Report	who(C) -H
Add†	similar to mkuser(ADM)
Delete†	similar to rmuser(ADM)
Modify→	
Username†	similar to pwcheck(C)
Passwd	passwd(C)
Shell†	similar to chsh(ADM)
Environment	edit <i>.profile</i> or <i>.login</i>
Group†	similar to pwcheck(C)



XENIX System Administrator's Guide

Communicate **mail(C)** or SCO Portfolio email

Backups→

Create † **sysadmin(ADM)**, **tar(C) -c**, **dd(C)**, or a utility similar to similar to **diskcp(C)**

Restore **tar(C) -x** or **sysadmin(ADM)**

Schedule **edit /usr/lib/sysadmin/schedule**

2

System→

Report→

Activity **ps(C)**

Users **who(C) -H**

Printers **lpstat(C)**

Disk **df(C) -v -i**

Network→

Micnet **netutil(ADM)**

UUCP **uustat(C)**

Xnet **xnstatus**

Messages→

Mail **mail(C) -u root**

Console **tail(C) /usr/adm/messages**

Software

custom(ADM)

shutdown(ADM)

Halt

shutdown(ADM)

Execute

run script in **/usr/lib/sysadm/local**

Configure→

Time **asktime(ADM)**

Autologout **idleout(ADM)**

Kernel **configure(ADM)**

Network→

UUCP **uinstall(ADM)**

Defaults→

Message **edit motd**

Checklist **edit /etc/checklist**

Other **edit any file in /etc/default**

International→

Individual **edit .profile** or **.login**

System **edit /etc/default/lang**

Display **mapchan(F)**

Keyboard **modifies /usr/lib/keyboard/keys**

Add→

HardDisk **mkdev(ADM) hd**

Printer **mkdev(ADM) lp**

Software	custom(ADM)
Card_Serial	mkdev(ADM) serial
Delete→	
HardDisk	mkdev(ADM) hd
Printer	mkdev(ADM) lp
Software	custom(ADM)

Filesystems→

Check	fsck(ADM)
Add	mkdev(ADM) fs
Mount	mount(ADM)
Unmount	umount(ADM)

Media→

List	tar(C) -t
Extract	tar(C) -x
Archive	tar(C) -c
Format	format(C) or dos(C) : dosformat
Duplicate†	similar to diskcp(C)
Tapedump	tapedump(C)

†Calls a sysadmsh function that is similar to the named XENIX command.



Chapter 3

Starting and

Stopping the System

- 3.1 Introduction 3-1
- 3.2 Starting the System 3-1
 - 3.2.1 Loading the Operating System 3-1
 - 3.2.2 Cleaning the File System 3-2
 - 3.2.3 Choosing the Mode of System Operation 3-2
- 3.3 Logging In As the Super User 3-3
- 3.4 Stopping the System 3-4
 - 3.4.1 Using the shutdown Command 3-4
 - 3.4.2 Using the haltsys Command 3-5



3.1 Introduction

This chapter explains how to start and stop the XENIX system. It also explains how to log in as the super user (root).

3.2 Starting the System

Starting a XENIX system requires more than just turning on the power. You must also perform a series of steps to initialize the system for operation. Starting the system requires:

- loading the operating system,
- cleaning the filesystem (if the system was improperly stopped), and
- choosing the mode of system operation.

3

The following sections describe each of these procedures.

3.2.1 Loading the Operating System

The first step in starting the system is to load the operating system from the computer's hard disk. Follow these steps:

1. Turn on power to the computer and hard disk. The computer loads the XENIX bootstrap program and displays the message:

```
Boot
:
```

2. Press the **RETURN** key. The bootstrap program loads the XENIX operating system.

When the system is loaded, it displays information about itself and checks to see if the "root filesystem" (i.e., all files and directories) is in order and not corrupted. If a filesystem is uncorrupted and in good order, it is called "clean." If the root filesystem is clean, you may choose the mode of operation. If not, the system requires you to clean the filesystem before choosing.

3.2.2 Cleaning the File System

You must clean the filesystem if the system displays the message:

```
Proceed with cleaning (y or n)?
```

This message is displayed only if the system was not stopped properly, as described in the section “Stopping the System” later in this chapter. The XENIX operating system requires a clean filesystem to work properly. If the above message does not appear, your filesystem is clean and ready to use.

To clean the filesystem, enter `y` (for “yes”) and press the **RETURN** key. The XENIX utility `fsck(ADM)` cleans the filesystem, repairing damaged files or deleting files that cannot be repaired. It reports on its progress as each step is completed. At some point, you may be asked if you wish to salvage a file. Always answer by entering “`y`” or “`n`” and pressing the **RETURN** key. For an explanation of how `fsck` works, see the “Filesystem Integrity” section of the “Using Filesystems” chapter in this guide.

When cleaning is complete, the system asks you to choose the mode of operation.

3.2.3 Choosing the Mode of System Operation

You may choose the mode of XENIX operation as soon as you see the message:

```
Type CONTROL-d to continue with normal startup,  
(or give the root password for system maintenance):
```

The system has two modes: *normal operation* and *system maintenance* mode. Normal operation is for ordinary work on the system. This is the mode that allows multiple users to log in and begin work. System maintenance mode is reserved for work to be done by the system administrator. It does not allow multiple users.

To choose normal operation, press **Ctrl-d**. The system displays a startup message and executes commands found in the command file */etc/rc* described in the chapter “Solving System Problems.” Next, the system displays the “login:” prompt. You may then log in as a normal user, as described in the “Logging In” chapter of the *XENIX Tutorial*, or as the super user, as described in the next section.

To choose system maintenance mode, enter the super user password (also called the “root password”) and press **RETURN**. The system displays the message of the day and the super user prompt (#). The commands in the */etc/rc* file are not executed. (Choose system maintenance mode only if you must do system maintenance work that requires all other users to be off the system.) When you log out of system maintenance mode using **Ctrl-d**, the system automatically enters normal operation.



To go from normal operation to system maintenance mode, log in as root and give the following command to shutdown the system, reboot and enter maintenance mode:

```
/etc/shutdown 2
```

△ **sysadmsh** users select: System→Halt

3.3 Logging In As the Super User

Many system maintenance tasks, when performed during normal operation, require you to log in as the super user. For example, you must be logged in as the super user to stop the system.

In order to log in as the super user, you must know the super user password. You also need to see the “login:” message on your terminal’s screen. If you do not see this message, press **Ctrl-d** until it appears.

To log in as the super user, follow these steps:

1. When you see the “login:” message, enter the super user’s login name:

```
root
```

and press the **RETURN** key. The system prompts you for the super user’s password.

2. Enter the super user’s password and press the **RETURN** key. The system does not display the password as you enter it, so enter each keystroke carefully.

The system opens the super user account and displays the message of the day and the super user prompt (#).

Take special care when you are logged in as the super user. In particular, you should be careful when deleting or modifying files or directories. This is because the super user has unlimited access to all files; it is possible to remove or modify a file that is vital to the system. Avoid using wildcard designators in filenames and keep track of your current working directory.

You can leave the super user account at any time by pressing **Ctrl-d**.

3

3.4 Stopping the System

Stopping the XENIX system requires more than just turning off the computer. You must prepare the system for stopping by using either the **shutdown** or the **haltsys** command. The following sections describe each command.

3.4.1 Using the shutdown Command

The **shutdown** command is the normal way to stop the system and should be used whenever the system is in normal operation mode. It warns other users that the system is about to be stopped and gives them an opportunity to finish their work.

To stop the system with the **shutdown** command, follow these steps:

1. Log in as the super user. See the section "Logging in as Super User" in this chapter. The system opens the super user account and displays the message of the day and the super user prompt.
2. Enter:

```
/etc/shutdown
```

```
Δ sysadmsh users select: System→Halt
```

and press the **RETURN** key. The system loads the command which in turn prompts you for the number of minutes you wish to have elapse before the computer stops:

```
Minutes till shutdown? (0-15):
```

3. Enter a number from 0 to 15 and press **RETURN**. The system displays a warning message at each terminal, asking logged-in users to finish their work and log out. As soon as all users are logged out or the specified time has elapsed, the system closes all accounts and displays the following message:

```
** Normal System Shutdown **  
  
** Safe to Power Off **  
-or-  
** Press Any Key to Reboot **
```



4. Turn off the computer or press any key to reboot the system.

3.4.2 Using the **haltsys** Command

The **haltsys** command halts the system immediately. This command should be used only when no other users are on the system. If there are any users logged into the system when the **haltsys** command is given, they are logged out and their work in progress is lost.

To stop the system with the **haltsys** command, follow these steps:

1. Log in as the super user. The system opens the super user account and displays the message of the day and the super user prompt.

2. Enter:

/etc/haltsys

and press the **RETURN** key. The system displays the following message:

```
** Normal System Shutdown **  
  
** Safe to Power Off **  
    -or-  
  
** Press Any Key to Reboot **
```



3

3. Turn off the computer, or press any key to reboot the system.

Chapter 4

Using Filesystems

- 4.1 Introduction 4-1
- 4.2 What is a Filesystem? 4-1
- 4.3 Adding a Second Hard Disk 4-2
 - 4.3.1 Mounting the New Filesystem 4-11
 - 4.3.2 Moving User Accounts Off the Primary Hard Disk 4-12
- 4.4 Maintaining Free Space In Filesystems 4-14
 - 4.4.1 Strategies for Maintaining Free Space 4-15
 - 4.4.2 Displaying Free Space 4-16
 - 4.4.3 Sending a System-Wide Message 4-16
 - 4.4.4 Displaying Disk Usage 4-17
 - 4.4.5 Displaying Blocks by Owner 4-17
 - 4.4.6 Mailing a Message to a User 4-18
 - 4.4.7 Locating Files 4-18
 - 4.4.8 Locating *core* and Temporary Files 4-19
 - 4.4.9 Clearing Log Files 4-19
 - 4.4.10 Removing and Restoring a Filesystem 4-20
 - 4.4.11 Expanding the Filesystem 4-20
- 4.5 Filesystems and Large Directories 4-21
- 4.6 Changing/Adding Filesystems on the Primary Hard Disk 4-21
- 4.7 Filesystem Integrity 4-21
 - 4.7.1 How XENIX Maintains Files 4-22
 - 4.7.2 How XENIX Maintains Filesystems 4-23
 - 4.7.3 Causes of Filesystem Corruption 4-24
 - 4.7.4 Rules For Checking Filesystems 4-24
 - 4.7.5 Repairing a Filesystem with *fsck* 4-25
 - 4.7.6 Summary of *fsck* Phases 4-26
 - 4.7.7 Automatic Filesystem Check 4-27



4.1 Introduction

This chapter describes one of the most important responsibilities of a system administrator: creating and maintaining filesystems. Included is a procedure describing the most common filesystem addition: adding a second hard disk, including an option to relocate user accounts to the new filesystem. In addition, strategies for maintaining free space are discussed. The concept of “filesystem integrity” is introduced, and how XENIX repairs damaged filesystems. For information on file permissions and other security considerations, see the “Maintaining System Security” chapter in this Guide.

4.2 What is a Filesystem?

A filesystem is a distinct division of the operating system, consisting of files, directories and the information needed to locate and access them. A filesystem can be thought of as a structure that directories and files are constructed upon.



Each XENIX system has at least one filesystem on the primary hard disk. This filesystem is called “the root file system” and is represented by the symbol “/”. The root filesystem contains the programs and directories that comprise the XENIX operating system. Typically, the root filesystem includes all the user directories as well. The primary hard disk can also be divided into more than one filesystem, as described in the “Installation Procedure” of the *XENIX Installation Guide*; one of the most common divisions is the */u* filesystem, used to isolate user accounts from the root filesystem. (For more details on these filesystems, see “Planning Your Disk Layout” in the “Installation Procedure” chapter of the *Installation Guide*.)

A XENIX system may also have other filesystems that contain special directories and application programs. Dividing the primary hard disk into multiple filesystems is done to protect the data and make maintenance easier. Adding still more filesystems by installing other hard disks expands the system storage space. New filesystems can be specifically created by the system administrator, then “attached” (mounted) and “detached” (unmounted) to the system when needed, the same way that a floppy disk is accessed. The next section describes how to add a new filesystem and, if desired, change the location of user accounts to the new disk. This is done without affecting the present configuration of the primary hard disk. (To change the current organization of filesystems on the primary hard disk, see “Changing/Adding Filesystems on the Primary Hard Disk.”)

4.3 Adding a Second Hard Disk

You can give the system extra room for storing user files and directories by adding a hard disk to the system. This is often the only remedy for a system that has one hard disk and suffers from chronic lack of space. See the *Release Notes* for a list of hard disks compatible with the current XENIX release.

You can only have one disk controller card. Software support is now provided for hard disks that do not have matching entries in the ROM tables. Switch settings on the disk controller card may need to be changed. Check your hardware manual for the hard disk drive and the computer for this information.

Before adding the new disk, you must know how to connect it to the computer. Connecting the hard disk is explained in the hardware manual provided with the disk.



Make sure the second drive is formatted and passes the manufacturer diagnostics before running XENIX. If it does not pass the diagnostic tests, you cannot use it with XENIX.

Note

In the steps outlined below, you are prompted to respond to a variety of prompts. Although it is not always designated in this documentation, remember to press **RETURN** (or **ENTER**) after you have typed each response. XENIX waits indefinitely until this is done.

These are the steps to add another hard disk with one XENIX filesystem and no DOS area:

1. Connect the hard disk, then boot the system and enter system maintenance mode.
2. When you are in system maintenance mode, enter:

```
/etc/mkdev hd
```

```
Δ sysadmsh users select: System→Add→HardDisk
```


3. The first utility invoked by **hdinit** is **dkinit**, which sets parameters for non-standard hard disks. You see the menu:

```

Hard Disk Drive 1 Configuration
    1. Display current disk parameters
    2. Modify current disk parameters
    3. Select default disk parameters

Enter an option or 'q' to quit:

```

dkinit is primarily for unusual or non-standard disks. If you have a standard hard disk, one that is supported by your computer hardware or special mother board ROM, enter **q** followed by RETURN to continue the installation.

Entering **q** at this point selects the default parameters for your hard disk. Unless you know that your disk is non-standard, assume that it is standard and enter **q** to continue your installation with **fdisk(ADM)**. Skip to step 5.

If your disk is non-standard, you must enter in information that overrides the ROM disk configuration information, replacing it with the new information. If you are unsure of what parameters to enter for your non-standard disk, contact your disk manufacturer for this information.

4. If your disk is non-standard, **dkinit** operates as follows:

If you enter “1” or “2”, you see the following display:

Disk Parameters	Values
1. Cylinders	<i>value</i>
2. Heads	<i>value</i>
3. Write Reduce	<i>value</i>
4. Write Precomp	<i>value</i>
5. Ecc	<i>value</i>
6. Control	<i>value</i>
7. Landing Zone	<i>value</i>
8. Sectors/track	<i>value</i>

When you see the display, “*value*” is replaced with the default value for that variable.



If you entered a "1", you now see the first menu again. If you entered a "2", you are now prompted:

Enter a parameter to modify or 'q' to return to the main menu:

Enter any of "1" - "8" to change the disk parameters, or **q** to return to the previous menu.

Enter the new value or <RETURN> to use the existing value:

4

If you wish to change the value, enter a new value now or press RETURN to use the existing value.

5. After you finish changing the disk parameters, enter **q** to return to the main menu. Next, enter **q** again to save the changes you made. Exiting from **dkinit** by entering **q** overwrites any parameters you have changed with the new values. If you wish to restore the default parameters after making modifications, enter "3" from the first menu.

As part of the initialization process, you may partition the hard disk, using the **fdisk**(ADM) utility, to support both DOS and XENIX on the same hard disk, or you can allow XENIX to use the whole disk.

6. After a moment, an **fdisk** menu appears on the screen. You see this option list:

1. Display Partition Table
2. Use Entire Disk for XENIX
3. Create XENIX Partition
4. Activate Partition
5. Delete Partition

Enter your choice or 'q' to quit:

If you would like XENIX to occupy the whole disk, select option “2”. If your disk already has valid partitions, you will also see the following warning message:

```
Warning! All data on your disk will be lost!  
Do you wish to continue? (y/n)
```

Enter **y** and press RETURN only if you want XENIX to occupy the whole disk. This ensures that **fdisk** partitions the whole disk for XENIX.

If no partitions were previously installed on your hard disk, **fdisk** will allocate the entire disk for XENIX without displaying the above warning.

Note that **fdisk** reserves the first track for **masterboot** and the last cylinder of the hard disk for disk diagnostics. Thus your partition begins on track 1 instead of track 0 and it ends on track 1219 instead of track 1223 in this example. Other hard disks will have different numbers.

You see the partition table again, with the following changes:

```
Current Hard Disk Drive: /dev/rhd10
```

Partition	Status	Type	Start	End	Size
1	Active	XENIX	1	1219	1218

```
Total disk size: 1224 tracks (5 reserved for masterboot  
and diagnostics)
```

```
Press <RETURN> to continue.
```

After you press RETURN, you see the main **fdisk** menu, shown above. Note that the XENIX partition must be active before exiting **fdisk**. Type **q** to exit **fdisk** and continue with the installation.

For more information on having DOS and XENIX on your hard disk, see **fdisk(ADM)**.



If you had a large portion of the disk already allocated to DOS, you must run DOS to deallocate this area. See **fdisk**(ADM) for more information on sharing disks between DOS and XENIX. No matter what configuration you produce with **fdisk**, the active partition must be the XENIX partition when you are through.

7. Now you see a menu from the program **badtrk**(ADM). With the **badtrk** program, you can scan your hard disk for defective tracks. The program maps any flawed locations to good tracks elsewhere on the disk. It also creates a bad track table, which is a list of all the bad tracks on your hard disk.

The main **badtrk** menu looks like this:

4

- ```
1. Print Current Bad Track Table
2. Scan Disk (You may choose Read-Only or Destructive later)
3. Add Entries to Current Bad Track Table by Cylinder/Head Number
4. Add Entries to Current Bad Track Table by Sector Number
5. Delete Entries Individually from Current Bad Track Table
6. Delete All Entries from Bad Track Table
```

Please enter your choice or 'q' to quit:

Enter "2", then press RETURN.

8. You see the following submenu:

- ```
1. Scan entire XENIX partition
2. Scan a specified range of tracks
3. Scan a specified filesystem
```

Select option "1".

9. After you select the area you want scanned, you are given the choice:

- ```
1. Quick scan (approximately 7 megabytes/min)
2. Thorough scan (approximately 1 megabyte/min)
```

Select option '2'.

10. You are prompted:

```
Do you want this to be a destructive scan? (y/n)
```

Enter **y**. You are warned:

```
This will destroy the present contents of the region you are scanning.
Do you wish to continue? (y/n)
```

Enter **y** and press RETURN. You see the following message:

```
Scanning in progress, press 'q' to interrupt at any time.
```

11. After you respond to the above prompts, the program scans the active partition of the new disk for flaws. The larger your disk, the longer the scanning process takes, so a very large disk may take a while.

As **badtrk** scans the disk, it displays the number of each track it examines, and the percentage of the disk already scanned. Pressing the **q** key at any time interrupts the scan. If you press **q** to interrupt the scan you do not need to press RETURN. You are then prompted to continue scanning or to return to the main menu.




Whenever **badtrk** finds a defective track, it lists the location of that track using both the sector number and cylinder/head conventions.

12. **If your disk comes with a flaw map**, you should enter any flaws from it into the bad track table.

Because most disk flaws are marginal or intermittent, your disk's flaw map will probably list more bad tracks than the scanning process reveals. If so, you should now add these defective tracks to the bad track table.

Select either option "3" or option "4" depending upon the format of the flaw map furnished with your disk. Enter the defective tracks, one per line. If you make a mistake, enter **q** and press RETURN. When you see the main **badtrk** menu, select option "5" to delete a track.

- 
13. **If your disk is not furnished with a flaw map, or you are finished making changes to the bad track table**, enter **q** and press RETURN.
  14. The program displays the number of identified bad tracks. You are next prompted for the number of tracks to allocate as replacements for those tracks that are flawed. You should allocate at least as many as the recommended number. Enter the number or just press RETURN to use the recommended number that is displayed:

Enter the number of bad tracks to allocate space for  
(or press return to use the recommended value of *n*):

If you press RETURN and do not enter an alternate value, **badtrk** allocates the recommended number of tracks as replacements. This number is based on the number of bad tracks currently in the table, plus an allowance for tracks that may go bad in the future. If you ever exceed the number of allocated bad tracks, you must reinstall XENIX.

15. Next, **badtrk** prompts:

```
Do you want to update this device with the new table?
```

Enter **y** and press RETURN to save the changes. To correct any mistakes or otherwise alter the bad track table, enter **n**. Modify the bad track table to contain the desired entries, enter **q** at the main menu to return to the prompt displayed above, then enter **y** to update the device with the new table.

16. Next, you see:

```
Do you want to attempt to salvage any valid data
on the bad tracks? [may take a long time] (y/n)
```



Enter **n** and press RETURN.

17. Next, you see a prompt from **divvy**. The **divvy** program divides a partition into filesystems. You can create up to seven divisions on a single partition, and name them anything you like.

If the hard disk you are installing is 20 megabytes or larger, you will be prompted for the number of filesystems you want to create. Press **RETURN** to use the default value of one filesystem. Smaller hard disks automatically default to creating one filesystem.

You are next prompted for block control of your hard disk. You see:

```
Do you require block by block control over
the layout of the XENIX partition? (y/n)
```

Enter **'y'** and press RETURN. This allows you to create up to seven filesystems on a single XENIX partition, and assign specific names to whatever filesystems you create. You must enter **'y'** if you are prompted to create more than 1 filesystem in the previous prompt.

You see the main **divvy** menu and a display that shows you how your disk is divided:

| Name | New File System? | # | First Block | Last Block |
|------|------------------|---|-------------|------------|
|      | no               | 0 | 0           | 25302      |
|      | no               | 1 | -           | -          |
|      | no               | 2 | -           | -          |
|      | no               | 3 | -           | -          |
|      | no               | 4 | -           | -          |
|      | no               | 5 | -           | -          |
|      | no               | 6 | -           | -          |
| hdla | no,exists        | 7 | 0           | 25546      |

25303 blocks for divisions, 244 blocks reserved for bad tracks

n[ame] Name or rename a division.

c[reate] Create a new filesystem on this division.

p[revent] Prevent a new filesystem from being created...

s[tart] Start a division on a different block.

e[nd] End a division on a different block.

r[estore] Restore the original division table.

Please enter your choice or 'q' to quit:

Each row in the **divvy** table corresponds to a filesystem. When you first see the table, there may be one or more filesystems created. If your disk is larger than 20 megabytes, you will see the number of filesystems that you specified at the first **divvy** prompt. You can change the default size of these filesystems by using the "start" and "end" options from the main **divvy** menu. Note that filesystem boundaries must not overlap. For example, filesystem 0 cannot end on the block number where filesystem 1 begins.

When you first see the main **divvy** menu, the filesystems are not named. Use option 'n' to change the name of a filesystem. Filesystems can have any name you choose. For example, you could name a filesystem *u* (for "user").

Do not change the configuration of filesystem 7. It is reserved for internal use by XENIX.

Once you have defined the start and end points of your filesystems, be sure to use the "(c)reate" option for each filesystem to make the filesystems on the disk.



Exit from **divvy** by entering 'q'. The program may prompt whether to install the new partition table, return to the main menu or exit the program without installing partition table. Select option 'i' to install the partition table.

If you have a large filesystem, you may be prompted if you want a scratch device to be created for you. You should answer yes as the scratch device is very useful for **fsck**.

For more information, see **divvy(ADM)** in this guide.

### 4.3.1 Mounting the New Filesystem

To use an additional disk, or create a second, mounted filesystem, you must make a new filesystem and mount it onto your system. The **divvy n[ame]** menu option created the device node */dev/u*. The next step is to create the filesystem itself. Using the filesystem */u* as an example, this is how the actual filesystem is created:

```
mkdev fs /dev/u /u
```

△ **sysadmsh** users select: Filesystems→Add

This command does the following:

- Creates a directory */u* (also known as the mountpoint).
- Creates the *lost+found* directory (used by **fsck** to recover files if the filesystem is corrupted).
- Mounts the device (*/dev/u*) on */u*, cleans it, and unmounts it.
- Removes write permissions on the directory */u* for group and all other users except root.
- Creates files in the */u/lost+found* directory, then removes them. This allocates inodes for the directory, so that if the filesystem is corrupted and runs out of inodes, **fsck(ADM)** is still able to recover files.
- Adds the following line to */etc/checklist*:

```
/dev/u
```

- Prompts you to decide if the new filesystem is to be automatically mounted, checked, and cleaned at boot time or not.



## Note

Note that your new filesystem does not have to be called `/u`. You can name your filesystems and mount points as you choose. It is recommended that mount points and filesystems have the same name to avoid confusion.

---

To mount or unmount `/dev/u` on `/u`, use the following two commands respectively:

```
mount /dev/u /u
```

△ `sysadmsh` users select: Filesystems→Mount

4

```
umount /dev/u
```

△ `sysadmsh` users select: Filesystems→Unmount

Only the super-user can use the `mount` command. The system administrator can permit users to mount specific filesystems (with or without password protection) by using the `mnt(C)` command. (For more information, refer to “Permitting Users to Mount Filesystems” in the “Maintaining System Security” in this guide.)

### 4.3.2 Moving User Accounts Off the Primary Hard Disk

You can access the files on your new filesystem by first mounting it in the appropriate directory (we used `/u` as an example in the preceding section). After you mount the filesystem, all directories and files on it are usable just as any others on the system.

Extending the example set forth in the previous sections, let's assume that you want to move your user accounts to the new `/u` filesystem on your secondary hard disk. The first thing to do is to ensure that any new accounts you add to the system will be placed in the new location. The `mkuser(ADM)` command (used to create new user accounts) reads the default location for user accounts from the file `/etc/default/mkuser`. In particular, the variable “HOME” must be changed to reflect the new location.

Edit the file */etc/default/mkuser*. As distributed, the file has an entry that looks like this:

```
HOME=/usr
```

If you wish to use */u*, change the entry accordingly to the name of the directory where you want to place user accounts. Now, whenever you run the **mkuser** command to add a new user, that user account will be in */u*. Make certain that the */u* filesystem is mounted before you run the **mkuser** script, or the new user's directory will not be accessible when */u* is mounted.

If there are already users on the system, and you want to move their accounts to the new filesystem, you can use the **cp(C)** command to copy their accounts to the new filesystem. You must also change the users' entries in */etc/passwd* to reflect the new pathnames of their home directories.

Follow these steps to move user accounts from one filesystem to another:

1. Make sure the new filesystem is mounted and the account in question is not being used. Also, be sure you are logged in as root.
2. Change directories to the top of the current user account directory. If, for example, the user accounts are in */usr*, enter:

```
cd /usr
```

and press **RETURN**.

3. List the contents of this directory:

```
ls
```

You see a list of account names, for example:

```
alisonb dean jerrys sams
blf gregt lost+found tammyr
buckm jeffj pj vicki
```



4. Enter:

```
copy -orm /usr /u
```

and press RETURN.

5. When the **copy** command has finished, enter:

```
cd /u
```

and press RETURN. List the new contents of */u* to make sure all of the accounts have been copied correctly.

6. **After you are sure that all of the accounts have been completely copied**, you can remove the user accounts in the previous user filesystem by removing files, directories and aliases to that location.
7. Change the home directory for each user as listed in */etc/passwd*. An example entry in */etc/passwd* might be:

```
alb:CoHiKNs.:271:104:Al Berry:/usr/alb:/bin/csh
```

You see one such line for every user on your system. Change the field:

```
:/usr/alb:
```

to match the user's new home directory:

```
:/u/alb:
```

Do this for every user whose home directory has changed.

## 4.4 Maintaining Free Space In Filesystems

Filesystem maintenance, an important task of the system administrator, keeps the XENIX system running smoothly, keeps the filesystems clean, and ensures adequate space for all users. To maintain the filesystems, the system administrator must monitor the free space in each filesystem, and take corrective action whenever it gets too low.

This chapter explains the filesystem maintenance commands. These commands report how much space is used, locate seldom-used files, and remove or repair damaged files.

The XENIX system operates best when at least 15% of the space in each filesystem is free. In any system, the amount of free space depends on the

size of the disk containing the filesystem and the number of files on the disk. Since every disk has a fixed amount of space, it is important to control the number of files stored on the disk.

If a filesystem has less than 15% free space, system operation usually becomes sluggish. If no free space is available, the system stops any attempts to write to the filesystem. This means that the user's normal work on the computer (creating new files and expanding existing ones) stops.

The only remedy for a filesystem which has less than 15% free space is to delete one or more files from the filesystem. The following sections describe strategies for keeping the free space available.

### 4.4.1 Strategies for Maintaining Free Space

The system administrator should regularly check the amount of free space on all mounted filesystems and remind users to keep their directories free of unused files. You can remind users by including a reminder in the message of the day file */etc/motd*.



If the amount of free space slips below 15%, the system administrator should:

1. Send a system-wide message asking users to remove unused files.
2. Locate exceptionally large directories and files, and send mail to the owners asking them to remove unnecessary files.
3. Locate and remove temporary files and files named *core*.
4. Clear the contents of system log files.
5. Make a complete backup of the filesystem, remove all the files, and then restore them again from the backup.
6. If the system is chronically short of free space, it may be necessary to create and mount an additional filesystem.

The above suggestions are described in detail in the following sections.

### 4.4.2 Displaying Free Space

You can find out how much free space exists in a particular filesystem with the **df** (for “disk free”) command. This command displays the number of “blocks” available on the specific filesystem. A block is 512 characters (or bytes) of data.

The **df** command has the form:

```
df specialfile
```

△ **sysadmsh** users select: System→Report→Disk

*specialfile* can be the name of a XENIX special file corresponding to the disk drive containing the file system. If you do not give a special filename, then the free space of all normally mounted file systems is given.

4

For example, to display the free space of the root filesystem */dev/root*, enter:

```
df /dev/root
```

and press **RETURN**. The command displays the special filename and the number of free blocks. You can find the percentage of free space to total space on your system with the command:

```
df -v
```

### 4.4.3 Sending a System-Wide Message

If free space is low, you may send a message to all users on the system with the **wall** (for “write to all”) command. This command copies the messages you enter at your terminal to the terminals of all users currently logged in.

To send a message, enter:

```
wall
```

and press **RETURN**. Enter the message, pressing **RETURN** to start a new line if necessary. After you have entered the message, press **Ctrl-d**. The command displays the message on all terminals in the system. To leave the **wall** command, press **Ctrl-d**. This removes the link to other terminals.

#### 4.4.4 Displaying Disk Usage

You can display the number of blocks used within a directory by using the **du** command. This command is useful for finding excessively large directories and files.

The **du** command has the form:

```
du directory
```

The optional *directory* must be the name of a directory in a mounted filesystem. If you do not give a directory name, the command displays the number of blocks in the current directory.

For example, to display the number of blocks used in the directory */usr/johnd*, enter:

```
du /usr/johnd
```

and press **RETURN**. The command displays the name of each file and directory in the */usr/johnd* directory and the number of blocks used.



#### 4.4.5 Displaying Blocks by Owner

You can display a list of users and the number of blocks they own by using the **quot** (for “quota”) command. The command has the form:

```
quot specialfile
```

The *specialfile* must be the name of the special file corresponding to the disk drive containing the filesystem.

For example, to display the owners of files in the filesystem on the hard drive */dev/hd1*, enter:

```
quot /dev/hd1
```

and press **RETURN**. The command displays the users who have files in the filesystem and the numbers of blocks in these files.

### 4.4.6 Mailing a Message to a User

If a particular user has excessively large directories or files, you may send a personal message to the user with the **mail** command.

To begin sending a message through the mail, enter:

**mail** *login-name*

△ **sysadmsh** users select: Users→Communicate

and press **RETURN**. The *login-name* must be the login name of the recipient. To send a message, enter the message, press **RETURN**, and then press **Ctrl-d**. If the message has more than one line, press **RETURN** at the end of each line. The **mail** command copies the message to the user's mailbox, where it may also be used via the **mail** command. See the *XENIX User's Guide* for details.



### 4.4.7 Locating Files

You may locate all files with a specified name, size, date, owner, and/or last access date by using the **find** command. This command is useful for locating seldom-used and excessively large files.

The **find** command has the form:

**find** *directory parameters*

The *directory* must be the name of the first directory to be searched. (It will also search all directories within that directory.) The parameters are special names and values that tell the command what to search for. See **find** (C) in the *XENIX User's Reference* for complete details. The most useful *parameters* are:

**-name** *file*

**-atime** *number*

**-print**

The **-name** parameter causes the command to look for the specified *file*. The **-atime** parameter causes the command to search for files which have not been accessed for the *number* of days. The **-print** parameter causes the command to display the locations of any files it finds.



For example, to locate all files named *temp* in the directory */usr*, enter:

```
find /usr -name temp -print
```

and press **RETURN**. The command displays the locations of all files it finds.

#### 4.4.8 Locating *core* and Temporary Files

You can locate *core* and temporary files with the **find** command.

A *core* file contains a copy of a terminated program. The XENIX system sometimes creates such a file when a program causes an error from which it cannot recover. A temporary file contains data created as an intermediate step during execution of a program. This file may be left behind if a program contained an error or was prematurely stopped by the user. The name of a temporary file depends on the program that created it.

In most cases, the user has no use for either *core* or temporary files, and they can be safely removed.



When searching for *core* or temporary files, it is a good idea to search for files which have not been accessed for a reasonable period of time. For example, to find all *core* files in the */usr* directory that have not been accessed for a week, enter:

```
find /usr -name core -atime +7 -print
```

and press **RETURN**.

#### 4.4.9 Clearing Log Files

The XENIX system maintains a number of files, called log files, that contain information about system usage. When new information is generated, the system automatically appends this information to the end of the corresponding file, preserving the file's previous contents. This means the size of each file grows as new information is appended. Since the log files can rapidly become quite large, it is important to clear the files periodically by deleting their contents.

You can clear a log file by entering:

```
cat < /dev/null > filename
```

where *filename* is the full pathname of the log file you wish to clear. A log file normally receives information to be used by one and only one program, so its

name usually refers to that program. Similarly, the format of a file depends on the program that uses it.

In some cases, clearing a file affects the subsequent output of the corresponding program. For example, clearing the file `/etc/ddate` forces the next backup to be a periodic backup.

### 4.4.10 Removing and Restoring a Filesystem

If your system has been in use for some time, the constant creation and removal of files creates a situation called *disk fragmentation*. This means that the files in the filesystem are written in small pieces on the hard disk. A small amount of disk space is used when a file is written across more than one portion of the disk. You can recover filesystem space (generally about 5 to 10 percent) by first making a complete backup of all the files in the filesystem and then removing all the files from the hard disk and restoring them from the backup. To make a complete backup of your system files, read the “Backing Up Filesystems” chapter in this guide for instructions on how to backup and restore a filesystem. (Disk fragmentation is a performance issue; for more information refer to the “Tuning System Performance” chapter in this guide.)

Since the files are completely rewritten on the disk, each file is written in one piece and fragmentation is reduced. A small amount of space will be recovered. It is a good idea to perform this action about once a year on a heavily used system and less often on a lightly used system. Be certain that you have complete, accurate, and readable backups before you begin or your files may be lost.

### 4.4.11 Expanding the Filesystem

If free space is chronically low, it may be to your advantage to expand the system's storage capacity by adding a second hard disk as described earlier in this chapter. Once it is mounted, you may use this new filesystem for your work, or even copy user or system directories to it.

A chronic shortage of space usually results from having more users on the system than the current hard disk can reasonably handle, or having too many directories or files. In either case, creating a new filesystem allows some of the users and directories to be transferred from the hard disk, freeing a significant amount of space on the existing filesystem and improving system operation.

## 4.5 Filesystems and Large Directories

It is wise to avoid directories that are larger than necessary. You should be aware of several special sizes. A directory that contains entries for up to 30 files (plus the required `.` and `..`) fits in a single disk block and can be searched very efficiently. One that has up to 286 entries is still a small directory; anything larger is usually a disaster when used as a working directory. It is especially important to keep login directories small, preferably one block at most. Note that, as a rule, directories never shrink. This is very important to understand, because if your directory ever exceeds either the 30 or 286 thresholds, searches will be inefficient; furthermore, even if you delete files so that the number of files is less than either threshold, the system will still continue to treat the directory inefficiently.

## 4.6 Changing/Adding Filesystems on the Primary Hard Disk

It is always best to plan the layout of your hard disk in advance as described in the *XENIX Installation Guide*. If you decide to change the number of filesystems on your hard disk, then you must back up your system and reinstall as described in the “Reinstalling and Updating Your System” chapter of the *XENIX Installation Guide*. During the installation process, you will use “block-by-block” control to reapportion your disk space as desired. It is important to realize that you cannot use backups created by the **backup(C)** or **sysadmin(ADM)** utilities to restore your system. Backups created by these utilities cannot be used to restore filesystems that were larger than the filesystems you plan to restore them to. This is true even if the backed-up filesystem was not full. For example, if you backed up a twenty megabyte filesystem that was only 50 percent full, you still won’t be able to restore the backup volumes on a fifteen megabyte filesystem. The reinstallation chapter explains that you must use **tar(C)** to back up your system.

4

## 4.7 Filesystem Integrity

We have explained that a filesystem is a distinct division of the XENIX operating system. It is part of the job of the operating system to maintain the integrity of filesystem data. Actual loss of data is a rare occurrence; XENIX filesystems are very resistant to corruption. This is because a certain amount of redundancy exists in special structures that are invisible to the user. It is these structures that ensure filesystem integrity. For example, when the system suffers a power outage, very little information is lost. Any damage usually affects one or two files, making them inaccessible. 99 percent of the time, XENIX can repair any damage done to files. In very rare cases, damage causes the entire filesystem to become inaccessible.

## XENIX System Administrator's Guide

The XENIX system uses the **fsck** program to repair damaged filesystems. **fsck** (for “filesystem check”) checks the consistency of filesystems. In cases where the contents of a file are lost (rare), the only way to restore lost data is from filesystem backups. **fsck** is run automatically on the root filesystem at boot time. The **fsck** status messages look like this:

```
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Pathnames
** Phase 3 - Connectivity
** Phase 4 - Reference Counts
** Phase 5 - Check Free List
```

If the system terminated abnormally (power outage) you see other messages that may seem alarming:

```
FREE INODE COUNT WRONG IN SUPERBLK (FIX?)
```

4

In fact, this kind of message is routine when a system was not shut down properly, and you have only to enter **y** and **fsck** will continue its recovery work. This could be done without the system administrator's intervention, but it is generally better to know what is happening to a filesystem after a problem has occurred.

In order to discuss the idea of filesystem integrity and how **fsck** functions, it is useful to explain the structure that underlies the simple idea of files, directories and filesystems. Although it is not necessarily vital to understand the principles of file storage, it is helpful to know what the messages like the one above refer to so they will seem less mysterious. After reading this section, you will understand some of the most basic principles of the XENIX operating system. The section “Repairing a Filesystem with **fsck**” explains the simple mechanics of using the **fsck** command. The subsections that follow explain the filesystem structures that **fsck** deals with.

### 4.7.1 How XENIX Maintains Files

Each filesystem contains special structures that allow the operating system to access and maintain the files and data stored on the filesystem. It is the disruption and repair of these structures that we are concerned with.

The structure of a filesystem is based on the way that hard disks store data. Although the hard disk contains all the data used by the system, it is not stored in neat little locations that correspond to individual files. It is unlikely that you could point to a spot on a hard disk and truthfully say: “My file is stored right there on this part of the disk.” In fact, the data is probably scattered across the disk, and a sophisticated addressing scheme is used by the

operating system so that it can access each of the pieces that a file is broken into and present it to the user as a unit.

The reason that data is spread around is because the operating system doesn't really deal with files, but with units of data. To explain what this means, let's assume that a file is created and is actually stored on one part of the disk. Now, suppose you edit that file, and delete a few sentences here and there. That means you are using a little bit less disk space than when you started. This space amounts to a series of gaps in the area where your file was stored. Disk space is a precious commodity and is not wasted. Those small amounts of memory are then allocated to other files. Picture this process on a scale of hundreds of files with a dozen users and you have an idea of how files are maintained in XENIX. Because of the effectiveness of the algorithms (formulas) that the operating system uses, this process is remarkably efficient and trustworthy.

### 4.7.2 How XENIX Maintains Filesystems



A filesystem contains files and directories, which are represented by special structures called "inodes" and "data blocks" that make it possible for the operating system to create and keep track of them:

**Data blocks**      Blocks are units of data stored on the disk.

**Inodes**            Inodes can be thought of as a card from a library card catalog. Each inode contains information about a file, just as a card contains information about a book, including its location and other important information. One important point to remember is that an inode does not contain the name of a file; directories contain the actual names. Inodes contain the locations of all the data that makes up a file so the operating system can collect it all when needed.

Blocks are not just stored on the hard disk. In order to minimize seeking data on the hard disk, recently-used data blocks are held in a cache of special memory structures called buffers. It is these structures that make the operating system more efficient. Some information is always lost when an outage occurs because recently-changed data that has not yet been written to the disk, but this is very minor.

With a hard disk filled with data, inodes, directories, files, and blocks cached in memory, how does the operating system keep track of them? The answer is that all these structures maintain sufficient connectivity between files and directories to allow severed connections to be reconstructed.

One special data block, the “super” block, contains overall information about the filesystem, rather than where a particular piece of a file is located. The super block contains the information necessary to mount a filesystem and access its data. It contains the size of the filesystem, the number of free inodes, and information about free space available.

In the case of the super block, the information is read when the filesystem is mounted, maintained in memory and written back to the disk every thirty seconds (the command `ps -ale` will show a process called “update.” It is this process that writes to the super block.)

### 4.7.3 Causes of Filesystem Corruption

In the case of all the structures mentioned in this section, corruption can occur. This means that the data or the structures used to locate data can be damaged. This can occur for several reasons:

4

|                    |                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hardware Failure   | Hardware failures are rare. The best way of dealing with it is to be sure that recommended diagnostic and maintenance procedures are followed conscientiously.                              |
| Program Interrupts | It is possible that errors that cause a program to fail might result in the loss of some data. It is not easy to generalize about this because the range of possibilities is so large.      |
| Human Error        | While it may be painful to admit it, probably the greatest cause of filesystem corruption falls under this heading. Here are three rules that should be followed when managing filesystems. |

### 4.7.4 Rules For Checking Filesystems

1. ALWAYS check a filesystem before mounting it. Nothing complicates the problem of cleaning up a corrupt filesystem so much as allowing it to be used when it is bad.
2. NEVER remove a filesystem physically without first unmounting it.
3. ALWAYS use the `sync(C)` command before shutting the system down and before unmounting a filesystem. (The `sync` command writes data in the buffer cache back to the disk.)

Regular filesystem backups represent the best assurance of continued filesystem integrity.

#### 4.7.5 Repairing a Filesystem with `fsck`

You can repair a filesystem with the `fsck` command. A filesystem must be unmounted before running `fsck`. (The root filesystem can't be unmounted, so the system must be shut down first and brought up again in single-user (maintenance) mode.) `fsck` examines the various structures on the disk and attempts to reconcile them. Where possible, it reestablishes connections, resolves references, it "cleans" a filesystem.

The command has the form:

```
fsck specialfile
```

△ `sysadmsh` users select: Filesystem→Check

The *specialfile* must be the name of the special file corresponding to the device name of the filesystem.

For example, let's assume that you have brought up your system after a power failure and are in single-user mode. To check the filesystem `/u`, which is represented by the device `/dev/u`, you would enter:

```
fsck /dev/u
```

and press **RETURN**. The program checks the filesystem and reports on its progress with the following messages:

```
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Pathnames
** Phase 3 - Connectivity
** Phase 4 - Reference Counts
** Phase 5 - Check Free List
```

If a damaged file is found during any one of these phases, the command asks if it should be repaired or salvaged. Enter `y` to repair a damaged file. You should always allow the system to repair damaged files even if you have copies of the files elsewhere or intend to delete the damaged files.

Note that the `fsck` command deletes any file that it considers too damaged to be repaired. You may elect to have `fsck` either make the repair or not. The reason you might choose to have `fsck` ignore an inconsistency is that you judge the problem to be so severe that you want either to fix it yourself using




the **fsdb(ADM)** utility, or you plan to restore your system from backups. If you cannot use **fsdb**, you must allow **fsck** to resolve the inconsistencies or the filesystem may not be usable.

Note that you may need to run **fsck** several times before the entire filesystem is clean. For a complete list of error messages, see the **fsck(ADM)** manual page.

### 4.7.6 Summary of fsck Phases

**fsck** scans and examines each of the structures mentioned earlier. Each phase compares components and checks that these components agree with each other.

 Phase 1 checks the blocks and sizes. **fsck** reads the inode list to determine the sizes and locate the blocks used by each file. Inodes are checked for validity which includes checking inode type, zero link counts, inode sizes, and for bad or duplicate blocks. (Bad blocks are block values outside the boundary of a filesystem.) When **fsck** asks whether or not to clear an inode, this means to zero out the bad information in the node. This removes the file or directory that was associated with it. A duplicate block means that two files point to the same block on the disk. **fsck** attempts to find the original inode along with the duplicate for correction in Phase 2.

Phase 2 checks the path names. Files removed in Phase 1 must then have their directory entries removed. Phase 2 cleans up error conditions caused by improper inode status, out of range inode pointers, and directories that point to bad inodes as described earlier. For files with duplicate blocks found in Phase 1, **fsck** will want to remove both files (this is one of the few areas where system administrator intervention is useful).

Phase 3 checks for connectivity. Phase 2 removed directories that don't point to valid files. Phase 3 reconnects files that have been severed from the directory structure. Any files that are unreferenced but valid are placed in a special directory called *lost+found*. Because the directory has been severed, the name of the file is lost and a number is assigned to the file in *lost+found*.

Phase 4 checks the reference counts. **fsck** checks the link count of each entry that survives Phases 2 and 3. In some cases, files that were not pointed to under the directory structure but still have an inode can be relinked back to the file system in *lost+found*.



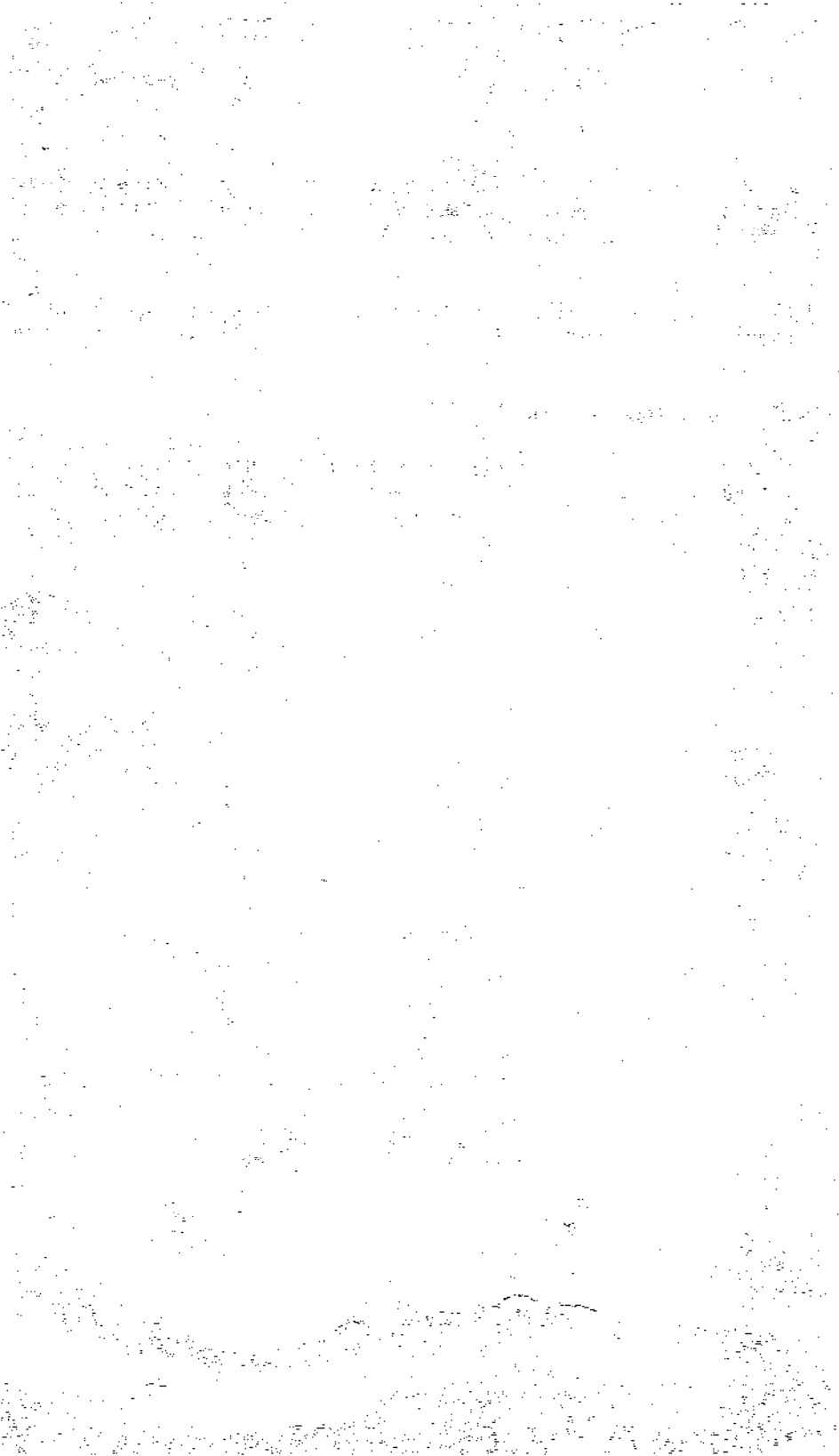
Phase 5 checks the free list. **fsck** examines the free-block list maintained by the filesystem and resolves the missing or unallocated blocks allocated or removed earlier. When an inconsistency is detected, **fsck** prompts to rebuild it.

Phase 6 salvages the free list. If specified in Phase 5, the system reconstructs a free block list from the altered filesystem.

### 4.7.7 Automatic Filesystem Check

The XENIX system sometimes requests a check of the filesystem when you first start it. This usually occurs after an improper shutdown (for example, after a power loss). The filesystem check repairs any files disrupted during the shutdown.





# Chapter 5

## Maintaining System Security

---

- 5.1 Introduction 5-1
- 5.2 General Security Practices 5-1
  - 5.2.1 Physical Security 5-2
  - 5.2.2 Access Security 5-2
  - 5.2.3 Protecting Special Files 5-3
- 5.3 Permissions 5-4
  - 5.3.1 Displaying Permissions 5-4
  - 5.3.2 Changing Permissions 5-6
  - 5.3.3 Changing the File Creation Mask 5-7
- 5.4 Managing File Ownership 5-8
  - 5.4.1 Changing User Ownership 5-8
  - 5.4.2 Changing Group Ownership 5-9
- 5.5 Changing a User's Password 5-9
- 5.6 Forcing a New Password 5-10
- 5.7 Adding Dial-in Password Protection 5-12
- 5.8 Permitting Users to Mount Filesystems 5-13
- 5.9 Using XENIX Accounting Features 5-14
  - 5.9.1 Starting Process Accounting 5-15
  - 5.9.2 Displaying Accounting Information 5-15



## 5.1 Introduction

This chapter outlines XENIX facilities and strategies for maintaining system security. System security can be reduced to two major components:

- Prevention of theft; of data on a system and of the system itself.
- Prevention of tampering; with data on a system and to the system itself.

Both aspects of security are important. Thefts can be prevented and tampering eliminated by the following:

- user education
- administrator education
- management education
- hardware confinement
- software support

Notice that software and hardware are at the bottom of the list. This is because the best software in the world is essentially useless if users give away their passwords. (Basic security is discussed from a user standpoint in “Keeping Your Account Secure” in “Logging In” chapter of the *XENIX Tutorial*.)

Users must be trained not to do such things as revealing their passwords or writing them down. The same is true for system administrators. Management must not only agree that theft- and tamper-prevention are important, but must support appropriate efforts to provide such prevention if system security is to be reliable.

This chapter describes controlling and recording user access to the files and directories on the system. It also introduces system security, permissions, password protection on various facilities, and process accounting.

## 5.2 General Security Practices

Every system, no matter what its size, should have some form of protection from unauthorized access to the computer, disks, and system files. The following sections suggest ways for a system administrator to protect the system.

## 5.2.1 Physical Security

You can protect the physical components of the computer, especially system disks, by taking these steps:

1. Keep your hardware secure and locked when not in use.
2. Organize and lock up all floppy disks when not in use. They should not be stored with the computer itself.
3. Keep disks away from magnetism, direct sunlight, and severe changes in temperature.
4. Do not use ball point pens to write labels on disks.
5. Make backup copies of all floppy disks. See the section "Copying Floppy Disks" in the "Using Floppy Disks and Tape Drives" chapter.
6. Keep an accurate and complete system log.

## 5

## 5.2.2 Access Security

You can protect the system from access by unauthorized individuals by taking these steps:

1. Remind users to log out of their accounts before leaving the terminal.
2. Remind users to use the **lock(C)** utility if they leave their terminals even for a short time.
3. Force logouts on idle terminals with the **idleout(ADM)** command.

Δ **sysadmsh** users select: System→Configure→Autologout

4. Discourage users from choosing passwords that are easy to guess. Passwords should be at least six characters long and include letters, numerals, and punctuation marks.
5. Passwords should not be people's or pet names, place names, or any word that can be found in */usr/dict/words*. Passwords should not be normal words spelled backwards or altered in any standard way. For example, "pig latin" forms of words should not be used. Passwords should never be normal words with an extra character or

numeral added. For example, “sally0” is a poor password choice.

6. Words from different languages or strings of short words make good password choices, provided they are not easily guessed. For example, a good password choice could be “gloCkenspiel.” Note that one of the central characters is capitalized and there is a period at the end of the word. The word is unusual and long enough to prevent easy guessing or decoding using a password testing program.
7. Passwords should be kept secret at all times. Passwords should never be written down, sent over electronic mail, or verbally communicated.
8. Force users to change their passwords regularly with the **pwadmin(ADM)** command.
9. Keep the super-user password secret from all but necessary people and change it regularly.
10. Restrict access to the XENIX distribution floppies.
11. Maintain permissions on important directories such as */etc*, */bin*, */usr*, and */dev*. These directories should be readable and executable by all users but only the super user should have write permission. Incorrect permissions on these directories are a serious lapse in system security. In addition, maintain the permissions on important system utilities such as **mkuser**, **shutdown**, **haltsys**, **restore**, **setuid**, and other utilities reserved for the super user. XENIX is distributed with the correct permissions set on all files and directories. If these permissions are not changed, they will not present a system security problem.

### 5.2.3 Protecting Special Files

You can prevent ordinary users from gaining direct access to the data and program files on the system’s hard and floppy disks by protecting the system’s special files. The XENIX special files, in the */dev* directory, are used primarily by the system to transfer data to and from the computer’s hard and floppy disks, as well as other devices, but can also be used by ordinary users to gain direct access to these devices.

Since direct access bypasses the system’s normal protection mechanisms and allows ordinary users to examine and change all files in the system, it is wise to protect the special files to ensure system security.

To protect the XENIX special files, log in as the super user and use the **chmod** command to set appropriate permissions. For example, to disallow any access by ordinary users, set the permissions of such special files as */dev/mem*, */dev/kmem*, */dev/root*, and */dev/usr* to read and write access for the user only. Note that you must not change the permissions for the */dev/tty* files.

### 5.3 Permissions

Permissions control access to all the files and directories in a XENIX system. In XENIX, ordinary users may access those files and directories for which they have permission. All other files and directories are inaccessible.

There are three different levels of permissions: user, group, and other. User permissions apply to the owner of the file; group permissions apply to users who have the same group ID as the owner; and other (public) permissions apply to all other users.

#### 5.3.1 Displaying Permissions



Any user can display the permission settings for all the files in a directory with the **l** (for "list directory") command. This command lists the permissions along with the name of the file owner, the size (in bytes), and the date and time the file was last changed. The command display has the following format:

```
-rw-rw---- 1 johnd group 11515 Nov 17 14:21 file1
```

The permissions are shown as a sequence of ten characters at the beginning of the display. The sequence is divided into four fields. The first field (the "type" field) has a single character, the other fields ("user," "group," and "other") have three characters each. The characters in the fields have the following meanings.

In the type field:

- d Indicates the item is a directory



- Indicates the item is an ordinary file
- b Indicates the item is a device special block I/O file
- c Indicates the item is a device special character I/O file

In the “user,” “group,” and “other” fields:

- r Indicates read permission. Read permission for a file means you may copy or display the file. Read permission for a directory means you may display the files in that directory.
- w Indicates write permission. Write permission for a file means you may change or modify the file. Write permission for a directory means you may create files or subdirectories within that directory.
- x Indicates execute permission (for ordinary files) or search permission (for directories). Execute permission for a file means you may invoke the file as you would a program. Execute permission for a directory means you may enter that directory with the `cd` command.
- Indicates no permission.



For example, the permissions:

```
-rwxrwxrwx
```

indicate an ordinary file with full read, write, and execute access for everyone (user, group, and other).

The permissions:

```
-rw-----
```

indicate an ordinary file with read and write access for the user only.

The permissions:

```
drwxr-x--x
```

indicate a directory with search access for everyone, read access for the user and group, and write access for only the user.

## XENIX System Administrator's Guide

When you create a file, the XENIX system automatically assigns the following permissions:

```
-rw-r--r--
```

This means that everyone may read the file, but only the user may write to it.

When you create a directory, the system automatically assigns the default permissions:

```
drwxr-xr-x
```

This means everyone may search and read the directory, but only the user may create and remove files and directories within it.

### 5.3.2 Changing Permissions

Any user can change the permissions of a file or a directory that they own with the **chmod** (for "change mode") command. This command requires that you tell it how to change the permissions of a specific file or directory. You do so by indicating which levels of permissions you wish to change (user "u", group "g", or other "o"), how you wish to change them (add "+" or remove "-"), and which permissions you wish to change (read "r", write "w", or execute "x"). For example, the pattern:

```
u+x
```

adds execute permission for the user. The pattern:

```
go-w
```

removes write permission for group and other.

The **chmod** command has the form:

```
chmod pattern file ...
```

△ **sysadmsh** users select: Dirs/File→Modify→Permissions

where *file* is the name of a file or directory. If more than one filename is given, they must be separated by spaces. For example, to change the permissions of the file “receivables” from “-rw-r--r--” to “-rw-----”, enter:

```
chmod go-r receivables
```

Press the **RETURN** key.

After using **chmod**, use the **l** command to check the results. If you have made a mistake, use **chmod** again to correct the mistake.

### 5.3.3 Changing the File Creation Mask

The file creation mask is a special number, kept by the system, that defines the permissions given to every file and directory created by a user. Initially, the mask has the value “022” which means every file receives the permissions:

```
-rw-r--r--
```

Every directory receives the permissions:

```
drwxr-xr-x
```

You can change the mask and the initial permissions your files and directories receive by using the **umask** command. Although executable as a shell command, **umask** is most useful when inserted into the user’s *.login* (for **cs**h users) or *.profile* (for Bourne shell users) file. This establishes individual **umask** defaults for each user.

The **umask** command has the form:

```
umask value
```

where *value* is a three-digit number. The three digits represent user, group, and other permissions, respectively. The value of a digit defines which permission is given as shown by the following table:

| Digit | Permission                                    |
|-------|-----------------------------------------------|
| 0     | Read and write (also execute for directories) |
| 1     | Read and write                                |
| 2     | Read (also execute for directories)           |
| 3     | Read                                          |
| 4     | Write (also execute for directories)          |
| 5     | Write                                         |
| 6     | Execute for directories                       |
| 7     | No permissions                                |

For example, the command:

```
umask 177
```

sets the file creation mask so that all files and directories initially have read and write permission for the user, and no permissions for all others.

## 5.4 Managing File Ownership

5

Whenever a file is created by a user, the system automatically assigns “user ownership” of that file to that user. This allows the creator to access the file according to the “user” permissions. The system also assigns a “group ownership” to the file. The group ownership defines which group may access the file according to the “group” permissions. The file’s group is the same as the creator’s group.

Only one user and one group may have ownership of a file at any one time. (These are the owner and group displayed by the **l** command.) However, you may change the ownership of a file by using the **chown** and **chgrp** commands.

### 5.4.1 Changing User Ownership

Any user can change the user ownership of a file they own with the **chown** command. The command has the form:

```
chown login-name file
```

△ **sysadmsh** users select: Files/Dirs → Modify → Ownership

where *login-name* is the name of the new user, and *file* is the name of the file or directory to be changed. For example, the command:

```
chown johnd projects.june
```

changes the current owner of the file *projects.june* to “johnd”.

The **chown** command is especially useful after changing the user ID of a user account.

### 5.4.2 Changing Group Ownership

You can change the group ownership of a file with the **chgrp** command.

The command has the form:

```
chgrp group-name file
```

△ **sysadmsh** users select: Files/Dirs→Modify→Ownership

where *group-name* is the name of a group given in the */etc/group* file and *file* is the name of the file you wish to change. For example, the command:

```
chgrp shipping projects.june
```

changes the group ownership of the file *projects.june* to the group named “shipping”.

The **chgrp** command is especially useful if you have changed the login group of a user.

### 5.5 Changing a User's Password

Normally, ordinary users can change the password of their own accounts with the **passwd** command (see the “Logging In” chapter of the *XENIX Tutorial*). Sometimes, however, it may be necessary for the super-user to change the password for a user. For example, if a password has been forgotten, and the user cannot get into the account to change it, it is necessary for the super-user to change the password. The super-user may change the password of any user (including the super-user account) with the **passwd** command.

To change a password, follow these steps:

1. Log in as the super-user.
2. Enter:

```
passwd login-name
```

(where *login-name* is the user's login name) and press the **RETURN** key. The command displays the message:

```
New password:
```

3. Enter the new password and press the **RETURN** key. The command does not display the password as you type it, so type carefully. The command then prompts you to enter the password again:

```
Retype new password:
```

4. Enter the password again and press the **RETURN** key.

To see how an ordinary user can change their own password with the **passwd** command, see the "Logging In" chapter of the *XENIX Tutorial*.

### 5.6 Forcing a New Password

From time to time, a user account may need a higher level of security than ordinary. Since the security of any account depends its password, it is important to keep the password as secret as possible. One way to provide greater security is to force users to change their passwords on a regular basis.

You can force users to change their passwords by using the **pwadmin** command. This command automatically dates each password and requires the user to provide a new password when the specified number of weeks have passed. The command also requires users to wait a minimum number of weeks before allowing them to restore their previous password. To use the **pwadmin** command, you must log in as the super-user.

You can enable password aging for a specified user by using the **-a** option. Enter:

```
pwadmin -a login-name
```

where *login-name* is the login name of a user. The user will then be required to wait a minimum number of weeks before he can change his password, and will be forced to change his password after a maximum number of weeks have elapsed. The **-a** option uses the default minimum and maximum values found in the */etc/default/passwd* file.

You can choose your own minimum and maximum number of weeks by using the **-min** and **-max** options. For example, a common pair of minimum and maximum values is 2 and 8. To set the minimum and maximum dates, enter:

```
pwadmin -min num -max num login-name
```

where *num* is a number in the range 0 to 63, and *login-name* is simply the login name of the user whose password you are administering. Note that the minimum and maximum cannot both be 0, and that the minimum must not be greater than the maximum.

If you are unsure of the current minimum and maximum values for a password, you can display them by entering:

```
pwadmin -d login-name
```

This command does not change the current values.

If you wish to force a user to change passwords immediately, enter:

```
pwadmin -f login-name
```

The user is prompted on his next login to supply a new password.



When a password no longer requires extra security, you can remove the current minimum and maximum values for the password by entering:

```
pwadmin -n login-name
```

The system will no longer prompt for changes.

For more information about password aging, see **pwadmin(ADM)** in this guide and **passwd(F)** in the *XENIX User's Reference*.

### 5.7 Adding Dial-in Password Protection

If desired, you can define special dial-in passwords on selected tty lines, requiring selected classes of users to input dial-in passwords. Logging information, including the last time of connection, can be stored for later use.

Specific dial-in lines that require passwords are defined in the file */etc/dialups*. The format is one tty device name per line, for example:

```
/dev/tty1A
/dev/tty5C
```

5

The actual dialup passwords are kept in the file */etc/d\_passwd*. The password format is the same used in */etc/passwd*, except only the first two fields (user name and encrypted password) are meaningful. The remaining fields, if present, are ignored.

The first field ("user name") in */etc/d\_passwd* is not a user name, but the name of a shell program (for example, */bin/sh*) used in */etc/passwd*. If the login shell of the user attempting to log in (on a tty line listed in */etc/dialups*) is listed in */etc/d\_passwd*, then the user is prompted for the dial-in password stored in */etc/d\_passwd*. (A shell name of "\*" in */etc/d\_passwd* specifies the default dialup password.)

A sample */etc/d\_passwd* file might be:

```
:<encrypted password>:Default dialup password
/usr/lib/uucp/uucico::UUCP dialup password (none)
/bin/rsh:<encrypted password>:Restricted shell user dialup password
```

The *<encrypted password>* cannot be a normal string. The following is the procedure for creating an encrypted password:



- Create a dummy user in */etc/passwd* (run **mkuser**(ADM) or edit the file directly).
- Run **passwd**(C) to set the password
- Edit the */etc/passwd* file and move that line from */etc/d\_passwd*. (If you use **mkuser**, be sure to run **rmuser**(ADM) so that the home directory and system mailbox of this dummy user are removed from the system.)

To enable time-of-login recording (and reporting of the time of last login at each login), create the log file using the following command:

```
touch /usr/adm/lastlog
```

In addition, if this file exists and the user is not currently logged in, the **finger**(C) utility will report the time of last login.

To establish the proper ownership and permissions on these files, enter the following commands:

```
chmod 644 /etc/d_passwd /etc/dialups /usr/adm/lastlog
chown bin /etc/d_passwd /etc/dialups /usr/adm/lastlog
chgrp bin /etc/d_passwd /etc/dialups /usr/adm/lastlog
```



The two files in */etc* can have greater restrictions if desired, but *lastlog* must be as specified above.

### 5.8 Permitting Users to Mount Filesystems

Only the super-user can use the **mount** command. However, the super-user can set up parameters to define which filesystems can be mounted by users with the **mnt**(C) command, including the use of an access password, if desired.

Each filesystem must have an entry in the file */etc/default/filesys*. The following is a sample set of entries:

```
bdev=/dev/root cdev=/dev/rroot mountdir=/ \
desc="The Root Filesystem" rcmount=no mount=no

bdev=/dev/u cdev=/dev/ru mountdir=/u rcmount=yes \
fsckflags=-y desc="The User Filesystem"

bdev=/dev/x cdev=/dev/rx mountdir=/x mount=yes \
rcmount=yes fsckflags=-y desc="The Extra Filesystem"
```


In simple terms, the entries above determine the following:

| <u>Filesystem</u> | <u>When Mounted</u> | <u>Can User Mount?</u> |
|-------------------|---------------------|------------------------|
| root              | boot time           | no                     |
| /u                | multiuser           | no                     |
| /x                | anytime             | yes                    |

If you wish to make any non-root filesystem mountable by users, simply add "mount=yes" to the entry for the given filesystem. In addition, when the **mnt** command is invoked without an argument (no filesystem name), the program will check all non-root filesystems to see if they can be mounted, and if so, mounts them. The option "mount=prompt" will ask the user if they want to mount each filesystem where a mount is permitted.

For more information on the **mnt** command, including a complete list of options, refer to the **mnt(C)** page in the *XENIX User's Reference*.

### 5.9 Using XENIX Accounting Features

 The XENIX system provides a set of commands that allows the system administrator to perform process accounting. Process accounting is a simple way to keep track of the amount of time each user spends on the system. The process accounting commands keep a record of the number of processes (i.e., programs) started by a user, how long each process lasts, and other information such as how often the process accesses I/O devices, and how big the process is in bytes.

Process accounting is helpful on systems where users are being charged for their access time, but it may also be used to develop a detailed record of system, command, and system resource usage.

There are several commands which may be used to do process accounting. Of these, the most useful are **accton(ADM)** and **acctcom(ADM)**. The **accton** command starts and stops process accounting. When invoked, the command copies pertinent information about each process to the file named */usr/adm/pacct*. The **acctcom** command is used to display this information. The command has several options for displaying different types of accounting information.

## 5.9.1 Starting Process Accounting

Process accounting can be started at any time, but is typically started when the system itself is started. You can start process accounting with the **accton** command. Enter:

```
accton /usr/adm/pacct
```

The command automatically creates a new file */usr/adm/pacct* and begins to copy process-accounting information to it. If the */usr/adm/pacct* file exists before starting **accton**, the file contents are deleted.

You can start process accounting automatically whenever you reboot the system. To do this, edit the system startup file */etc/rc*. Among other commands, there are several that start up process accounting, as well as back up the accounting log file */usr/adm/pacct*.

These commands are commented out but, if you remove the comment characters at the beginnings of those lines, the commands are executed every time you reboot the system. You must edit the following lines in */etc/rc*:

```
mv /usr/adm/pacct /usr/adm/opacct
> /usr/adm/pacct ; chmod 644 /usr/adm/pacct
[-x /etc/accton] && /etc/accton /usr/adm/pacct
```

The pound signs (#) in front of each line are comment symbols that cause the line to be ignored. Remove the pound signs from each line and close the file. Note that, when you start the system after editing */etc/rc*, the contents of the */usr/adm/pacct* file are saved in the file */usr/adm/opacct*, overwriting the contents of */usr/adm/pacct*.

## 5.9.2 Displaying Accounting Information

The **acctcom** command reads process-accounting information from the */usr/adm/pacct* file by default, then displays selected information on your terminal screen. The command usually displays basic accounting information, such as the process' program name, the name of the user who invoked the process, the start and stop times for the process, and the number of execution seconds in real time and CPU time. The command has several options that can be used to display selected information.

To display the average size of each process, enter:

```
acctcom
```

## XENIX System Administrator's Guide

The command displays the basic information plus the average size of each process.

To display basic accounting information about a specific command, enter:

```
acctcom -n command
```

where *command* is the name of the command you are interested in. The command responds by displaying each entry for the specified *command*. For example, to display each entry for the system command **units**, enter:

```
acctcom -n units
```

To display information about the number and size of input and output counts, enter:

```
acctcom -i
```

The command displays basic program information plus the numbers of characters and blocks transferred or read by each program.

5

To display information about a program's use of system resources, enter:

```
acctcom -h
```

The command displays the basic information plus the "use factor." This is a number generated and used by the system to determine how each process should be scheduled for execution. Processes with high use factors use a high percentage of the system resources and are therefore scheduled after processes with lower factors.

# Chapter 6

## Backing Up Filesystems

---

- 6.1 Introduction 6-1
- 6.2 Strategies for Backups Using `sysadmin` 6-1
  - 6.2.1 Using the backup Account for Backups 6-2
  - 6.2.2 Floppy Drive Backups and Large Systems 6-2
  - 6.2.3 Summary of Utilities Accessed 6-3
- 6.3 Preparations for Scheduled Backups 6-3
  - 6.3.1 Creating a Backup Schedule 6-3
  - 6.3.2 Labeling Your Backups 6-8
  - 6.3.3 Keeping a Log Book 6-9
- 6.4 Performing a Scheduled Backup 6-9
  - 6.4.1 Using Formatted Media 6-10
  - 6.4.2 Starting the Backup 6-10
- 6.5 Performing an Unscheduled Backup 6-13
- 6.6 Getting a Backup Listing 6-16
- 6.7 Restoring Individual Files or Directories From Backups 6-17
- 6.8 Restoring an Entire Filesystem 6-20
- 6.9 Editing `/etc/default/filesys` and `/etc/default/archive` 6-25
- 6.10 An Explanation of Backup Levels 6-27
  - 6.10.1 Principles of Incremental Backup Levels 6-27
  - 6.10.2 How the Default Schedule Works 6-28
  - 6.10.3 How Backups are Used to Restore a Filesystem 6-29



## 6.1 Introduction

The main task of a system administrator is to ensure the continued integrity of information stored on the system. Files and filesystems can be damaged and data lost in the following ways:

- Power interruptions (make certain you have a surge protector)
- Hardware failures (particularly the hard disk)
- User errors (accidental removal of important files).

The importance of having up-to-date backups cannot be overstated. If your system has a number of active users, backups require daily attention. It is difficult to estimate the magnitude of a simple loss of data until an accident occurs and several weeks or months of work is gone in an instant.

A filesystem backup is a copy, on storage media (floppy disks or tape) of the files in the root directory and other regularly mounted filesystems. (See the “Using Filesystems” chapter in this Guide for a discussion of filesystems.) A backup allows the system manager, when logged in as **backup**, to save a copy of the filesystem as it was at a specific time.

This chapter explains how use the **sysadmin**(ADM) utility to create backups of the root directory and other filesystems, and how to restore files from the backups. (Another utility used for simple backups, **tar**(C), is discussed extensively in “Making Backups” in the “Housekeeping” chapter of the the *XENIX Tutorial*.)



The tools discussed in this chapter present menus with simple options instead of the complex command lines used with the utilities **tar**(C), **backup**(C) and **restore**(C). The key to efficient backups is to save only what has changed from day to day, which (when used with **backup** and **restore**) normally requires extra bookkeeping.

## 6.2 Strategies for Backups Using sysadmin

As system administrator, you should familiarize yourself with this chapter and create a schedule as instructed. When this schedule is complete, you have only to insert a media volume and respond to a series of prompts to perform your daily backups.

The primary purpose of the **sysadmin** Filesystem Maintenance Menu is to provide a dependable schedule of filesystem backups for systems with many users and large filesystems. The program automatically locates modified files and copies them to backup media. If your system has many users and a large number of files that are modified daily, the “scheduled” backup option uses a predefined schedule to make regular backups. When the Filesystem Maintenance Menu is invoked, the program presents each task as a menu option. To perform a task, simply choose the appropriate option from the menu and supply any required information.

For backups of a more informal nature, **sysadmin** includes an option for “unscheduled” backups. This allows the system administrator to perform a single, complete backup of a filesystem. (Note this type of backup covers the entire filesystem, not just modified files, and may require a number of storage media volumes.) If you intend to rely on unscheduled backups, be sure and perform one at least once a month.

### 6.2.1 Using the backup Account for Backups

Always use the **backup** account whenever you make or restore backups. (You must be **root** to restore an entire filesystem.) An ordinary user cannot make backups because they do not have access permissions for all files. If backups are made as **root**, files may be accidentally destroyed because **root** has unrestricted permissions on every file on the system. The **backup** account solves this dilemma by having restricted **root** permissions. Logging in as **backup** takes you directly to the Filesystem Maintenance Menu.

The **backup** account already exists on your system; you set the password when the BACKUP package was installed.

### 6.2.2 Floppy Drive Backups and Large Systems

If your system has only a floppy drive, backups for large systems with several users can be time-consuming and use a great deal of media. A complete backup of a 20 megabyte filesystem requires *fifteen* 1.2 MB 96tpi diskettes, whereas a single 450 foot cartridge tape can store more than twice that amount. More importantly, diskettes require the presence of the operator to keep feeding floppies, whereas a single cartridge tape can be inserted and the operator need not remain by the system. If your system has a large number of users and just a floppy drive, you are advised to install a cartridge tape drive, or make complete system backups once per week and warn your users to make individual backups of their own files on a regular basis.



### 6.2.3 Summary of Utilities Accessed

**sysadmin** accesses several utilities during this process. You need not be familiar with them. However, should you wish to use advanced options not discussed in this chapter, you will need to know how they are used and which *XENIX Reference* pages to read. **sysadmin** uses the Filesystem Maintenance menu to access the following utilities:

- **fsphoto(ADM)**: main utility that controls the automated backup facilities.
- **fsave(ADM)**: the program that interacts with the user to perform the backup.
- **schedule(ADM)**: the backup database.
- **backup** and **restore(C)**: the actual backup utilities; each of the tools above (including **sysadmsh** and **sysadmin**) form the “user-friendly” layer that isolates the user from these programs.

### 6.3 Preparations for Scheduled Backups

The only mandatory requirement for scheduled backups is the creation of a backup schedule. In addition, it is recommended that the system administrator follow the optional procedures for labeling, storing and logging backups. A detailed explanation of backup levels is included at the end of this chapter in case it is necessary to design a more specialized schedule.



#### 6.3.1 Creating a Backup Schedule

The first step is to create a timetable for backups using the **schedule** file. The file is located in */usr/lib/sysadmin*. It contains all the data needed for the system to perform a system backup, including:

- The name of your site or machine
- The media type and drive to be used
- A precise schedule of filesystems to be backed up.

The sections that follow explain what changes should be made to the **schedule** file provided with your XENIX distribution.

## Edit the schedule File

You can edit the **schedule** file with any text editor; make certain you are logged in as **root**. For example, if you use the **vi(C)** editor, you would enter the following command to edit the **schedule** file:

```
vi /usr/lib/sysadmin/schedule
```

△ **sysadmsh** users select: Backups→Schedule

The subsections that follow explain the exact changes you need to make to this file.

```

SYSTEM BACKUP SCHEDULE
site mymachine

Media Entries
48 tpi 360K floppy 0
media /dev/rfd048ds9 k 360 format /dev/rfd048ds9
48 tpi 360K floppy 1
media /dev/rfd148ds9 k 360 format /dev/rfd148ds9
96 tpi 720K floppy 0
media /dev/rfd096ds9 k 720 format /dev/rfd096ds9
96 tpi 720K floppy 1
media /dev/rfd196ds9 k 720 format /dev/rfd196ds9
96 tpi 1.2 MB floppy 0
media /dev/rfd096ds15 k 1200 format /dev/rfd096ds15
96 tpi 1.2 MB floppy 1
media /dev/rfd196ds15 k 1200 format /dev/rfd196ds15
135 tpi 1.44 MB floppy 0
media /dev/rfd0135ds8 k 1440 format /dev/rfd0135ds8
135 tpi 1.44 MB floppy 1
media /dev/rfd1135ds8 k 1440 format /dev/rfd1135ds8

Cartridge tape 0
media /dev/rct0 d 20000 300 450 600 tape erase
Mini cartridge drive (10MB)
media /dev/rctmini k 8800 format /dev/rctmini
Mini cartridge drive (40MB)
media /dev/rctmini k 37500 format /dev/rctmini

9-track tape drive
media /dev/rmt0 d 1600 2400 1200 600

Backup Descriptor Table

Backup Vol. Save for Vitality Label
level size how long (importance) marker
0 - "1 year" critical "a red sticker"
1 - "4 months" necessary "a yellow sticker"
8 - "3 weeks" useful "a blue sticker"
9 - "1 week" precautionary none

Schedule Table

1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
Filesystem M T W T F M T W T F M T W T F M T W T F
/dev/rroot 0 x 9 x 9 8 x 9 x 9 1 x 9 x 9 8 x 9 x 9
/dev/zu 9 0 9 9 9 9 8 9 9 9 9 1 9 9 9 9 8 9 9 9

```



Figure 6-1: The schedule File

### Add the Name of Your Site or Machine

Simply change the *mymachine* entry at the top of the file to the name used in your */etc/systemid* file, or whatever name you wish.

### Select the Media Device that Matches Your Configuration

The default **schedule** file appears as in Figure 6-1. The 96tpi 1.2 Mega-byte floppy drive 0 is the default drive (reproduced below). The pound signs (#) are comment symbols used to “comment out” text so that it is ignored by the program. Note that the default drive is the only one without a comment symbol. If you plan to use a drive other than the default, put a comment symbol in front of the 96tpi drive and remove the comment symbol from in front of the drive you wish to use. The remaining drives should remain commented out.

```
96 tpi 720K floppy 1
media /dev/rfd196ds9 k 720 format /dev/rfd196ds9

96 tpi 1.2 MB floppy 0
media /dev/rfd096ds15 k 1200 format /dev/rfd096ds15

96 tpi 1.2 MB floppy 1
media /dev/rfd196ds15 k 1200 format /dev/rfd196ds15
```



6

### Edit the Backup Descriptor Table

Directly below the media drive lines is the Backup Descriptor table. This table (below) describes each backup level in terms of volume size, how long it is to be stored, how important it is, and how it is marked. The default entries should prove useful, but the volume size entries must be edited according to the type of media you are using.

If you are using floppy disks, leave the dashes in the “Vol. size” column as they are. This causes the backup program to take the volume size from the media entry for that device.

If you are using tapes or tape cartridges, replace each dash in the “Vol. size” column with the size (in feet) of the tape volume. If you are using tapes that are all the same size for each backup level, replace each with the size of the tape you’re using.

The last column contains label entries that are discussed in “Labeling Your Backups” later in this section.

| # | Backup level | Vol. size | Save for how long | Vitality (importance) | Label marker       |
|---|--------------|-----------|-------------------|-----------------------|--------------------|
| 0 | -            | -         | "1 year"          | critical              | "a red sticker"    |
| 1 | -            | -         | "4 months"        | necessary             | "a yellow sticker" |
| 8 | -            | -         | "3 weeks"         | useful                | "a blue sticker"   |
| 9 | -            | -         | "1 week"          | precautionary         | none               |

Figure 6-3: Backup Descriptor Table

### Edit the Backup Schedule Table

The default schedule assumes that daily backups will be done. A precise understanding of backup levels is not critical to using the schedule. Level 0 is the lowest level backup. It backs up everything on the filesystem, while 1, 8, and 9 each back up only the files that have changed relative to the last lower-level backup. (For a complete discussion, see “An Explanation of Backup Levels” at the end of this chapter.) Note that there is a level 0 done once per month for the root filesystem and twice per month for the */u* filesystem. This is because the */u* filesystem (user accounts) changes much more frequently than the root filesystem, which contains the system files.

If you do not have a */u* filesystem, then your user accounts are located in the root filesystem (in */usr*). If this is so, delete the entire line for */dev/rrroot* and change */dev/ru* to */dev/rrroot*. This must be done so that the entire hard disk will be backed-up.



| # | Filesystem         | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 |
|---|--------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # | Filesystem         | M | T | W | T | F | M | T | W | T | F | M | T | W | T | F | M | T | W | T | F |
|   | <i>/dev/rrroot</i> | 0 | x | 9 | x | 9 | 8 | x | 9 | x | 9 | 1 | x | 9 | x | 9 | 8 | x | 9 | x | 9 |
|   | <i>/dev/ru</i>     | 9 | 0 | 9 | 9 | 9 | 9 | 8 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 8 | 9 | 9 | 9 |

Figure 6-4: Backup Schedule Table

Note that the Monday-Friday notation can be misleading; if a backup is postponed or unsuccessful (because of bad media, for example) then that same level backup is attempted again at the next scheduled backup. This offsets the schedule, but does not alter the established sequence of backups. The numbered scale of 1-0 above M-F is more accurate, but less useful to people, who work in day and week units.

In addition, if you add lines for other filesystems, you should take care not to schedule two level 0 backups of large filesystems on the same day; the process will be lengthy and may slow your machine significantly.

## 6.3.2 Labeling Your Backups


It is important to label your backup tapes with meaningful and accurate information. If your backups consist of a pile of haphazardly labeled tapes, it will be difficult to locate data at a later date.

The following is a suggested format for media labels:

|                       |                       |               |
|-----------------------|-----------------------|---------------|
| Name of computer      | Backup level          | Date made     |
|                       | Filesystem Name       |               |
|                       | save until date       |               |
|                       | # of blocks on volume |               |
| Name of backup person |                       | volume # of # |

The date on the label, and the date from which you calculate the "save until" date, should be the date of the business day covered by the backup. This is to avoid confusion if it becomes necessary to restore information from this tape.

You may have noticed that the **schedule** file has a proposed color-coding scheme for easy reference:



| # | Backup level | Vol. size | Save for how long | Vitality (importance) | Label marker       |
|---|--------------|-----------|-------------------|-----------------------|--------------------|
| 0 |              | -         | "1 year"          | critical              | "a red sticker"    |
| 1 |              | -         | "4 months"        | necessary             | "a yellow sticker" |
| 8 |              | -         | "3 weeks"         | useful                | "a blue sticker"   |
| 9 |              | -         | "1 week"          | precautionary         | "none"             |

Figure 6-5: Backup Labeling Scheme

If there is more than one tape for a single backup, mark the date label on each volume to indicate the volume number and number of volumes, such as "1 of 2" and "2 of 2" for a two volume backup.

Finally, place a label on the side of the box or enclosure marked with the name of the computer, the filesystem, and the backup level completed.

### 6.3.3 Keeping a Log Book

It is recommended that a written log book be maintained for each computer. In addition to maintenance information (such as when breakdowns occur and what was done about it), you should record the following information:

|                   |                                                                                                                                                                                                                                                                                                                                 |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Date              | Just as with the tape label, this date should be the last day covered by the backup.                                                                                                                                                                                                                                            |
| Filesystem        | The name of the device backed-up on the current tape.                                                                                                                                                                                                                                                                           |
| Backup level      | The backup level of the current tape.                                                                                                                                                                                                                                                                                           |
| #Vols             | Number of tape volumes.                                                                                                                                                                                                                                                                                                         |
| #Blocks           | Number of tape blocks. This information is output when the backup is completed. (Pay attention to this figure; if the number of blocks for a level 9 backup consistently exceeds four digits for a particular filesystem, then you should probably increase the frequency of backups for that filesystem to lessen the burden.) |
| Start/finish time | (Optional.) The time from the start of a backup of a filesystem until the last error check is completed. The times are displayed after the backup is finished. The finish time will often be inaccurate, since you may be out of the room when the backup finishes, and the machine sits idle before you return.                |



If there are problems with the backup, record these in the log book as well, including any error messages that come to the screen.

### 6.4 Performing a Scheduled Backup

This section describes how to perform a backup using a defined schedule. Do not attempt this until you have edited (or at least examined) the schedule file to make certain that it suits your needs.

The system administrator should schedule backups at times when few (if any) users are on the system. This ensures that the most recent version of each file is copied correctly.

A regular schedule of backups requires a good supply of media and adequate storage for them. Level 0 backups should be saved indefinitely; lesser backups should be saved at least two weeks. Media volumes should be properly labeled with the date of the backup and the names of the files and directories contained in the backup. After a backup has expired, the media may be used to create new backups.

### 6.4.1 Using Formatted Media

If you use media that requires formatting, such as floppy disks or mini tape cartridges, you may wish to format several volumes before you begin. The exact number of volumes depends on the number and size of files to be backed up. For details on how to format your media, see the "Using Floppy Disks and Tape Drives" chapter in this guide. You also have the option to do formatting from the **sysadmin** program. (Note that rctmini tape cartridges take a very long time to format.)

### 6.4.2 Starting the Backup

To run your scheduled backup, follow these steps:

1. Log in as **backup**. After any login messages are displayed, you are taken directly to the Filesystem Maintenance Menu:

6

1. Perform a scheduled backup
2. Perform an unscheduled backup
3. List the contents of an archive
4. Restore backed up file(s)
5. Restore an entire filesystem
6. Check backup archive integrity

Enter an option or enter q to quit:

2. Enter "1" for a daily backup and press **RETURN**.



3. A menu is displayed that looks like the following:

```
Level 0 backup of filesystem /dev/rroot, 22 Sep 1988
 tape size: 450 feet [or Kb]
 tape drive: /dev/rct0
This tape will be saved for 1 year, and is critical.

M)ounted volume, P)ostpone, C)heck or F)ormat volumes,
 R) Retension or H)elp:
```

The media type displayed is the one entered in the **schedule** file. Load a volume, tape or disk, into the selected drive. Enter "m" to tell the program the volume is mounted, and press **RETURN**.

4. The system displays the current date and the date of the last backup:

```
Backup date= the epoch
Backing up /dev/rroot to /dev/rct0
```

(If you have not performed a backup previously, the system has no last backup date recorded and "the epoch" is displayed as the previous backup date.) An estimate of the number of volumes necessary is displayed. The system then begins to copy files to the drive. If a volume runs out of space, the program displays the message:

```
Please insert new volume, then press <RETURN>
```



## Note

If you are using 5.25-inch floppies for your backups, make certain you close the floppy door before pressing **RETURN**, or the entire backup will be aborted and you will have to start over.

---

Remove the present volume, insert a new volume, then press **RETURN**. The program continues to copy files to the new volume. Repeat this step until the program displays the message:

DONE

5. After the backup is complete, you are advised to select "c" from the main menu to perform a check of the format. This is not a check of the format used in formatted media, but a confirmation of the presence of header information that should be present on each volume of a backup. The following message is displayed:

```
Check critical volumes for format errors
M)ounted first volume, S)kip format check, or H)elp:
```

If you wish to have the volume checked, insert the first volume of the backup, select "m" and press **RETURN**. If you wish to skip checking the volume format, and continue on to the read error check, select "s" and press **RETURN**.

---

## Note

During verification, you may see messages indicating that a file on the hard disk differs from the backup. This is because certain files can change during the short interval between backup and verification. These messages do not affect the backup and can be ignored.

---

6. Every volume should be checked for read errors, and they must be checked in first-to-last order. (The **restore(C)** utility is used to check for errors during reading.) If an error occurs, the backup is

declared unsuccessful and is retried from the beginning. The menu appears as follows:

```
M)ounted which volume, E)rror on previous volume, D)one,
S)kip checks, or H)elp:
```

7. If you wish to check each of the backup volumes, you should insert them in order, enter “m” and press **RETURN**. If you wish to skip the check, enter “s” and press **RETURN**.
8. If an error occurs on the last volume checked, discard the suspect volume and start the backup again by entering “e” and pressing **RETURN**.
9. When all volumes have been checked and no errors have occurred, enter “d” and press **RETURN** to exit the program

After the backup has been successfully performed, instructions are given on how to label the volumes.

### 6.5 Performing an Unscheduled Backup

You can create backups on tape or disk. If you use media that requires formatting, such as floppy disks, you may wish to format several volumes before you begin. The exact number of volumes depends on the number and size of files to be backed up. For details on how to format media, see the “Using Floppy Disks and Tape Drives” chapter in this guide. You also have the option to do formatting from within the Filesystem Maintenance Menu.



To create a backup, follow these steps:

1. Log in as **backup**. The Filesystem Maintenance Menu is displayed:

```
1. Perform a scheduled backup
2. Perform an unscheduled backup
3. List the contents of an archive
4. Restore backed up file(s)
5. Restore an entire filesystem
6. Check backup archive integrity
```

Enter an option or enter q to quit:

2. Enter "2" for an unscheduled backup and press **RETURN**.
3. You see the following menu:

```
1. / - the root filesystem
2. /u -
3. Other
```

Select a filesystem to backup  
or enter q to return to the main menu:

The menu lists all filesystems found in the file */etc/default/filesys* (discussed later in "Editing */etc/default/filesys* and */etc/default/archive*"). Select the number of the filesystem you wish to back up and press **RETURN**. (If you wish to enter a filesystem that is not listed in */etc/default/filesys*, select "Other" and you will be prompted for the name.)

4. Next, you are asked to select the media device to be used:

- ```
1. Floppy Drive 0 (48dsdd)
2. Floppy Drive 1 (48dsdd)
3. Floppy Drive 0 (96dshd)
4. Floppy Drive 1 (96dshd)
5. Floppy Drive 0 (96dsdd)
6. Floppy Drive 1 (96dsdd)
7. Cartridge Drive (300 ft tape)
8. Cartridge Drive (450 ft tape)
9. Cartridge Drive (600 ft tape)
10. Mini-Cartridge Drive (10MB)
11. Mini-Cartridge Drive (40MB)
12. Other
```

Select an archive device,
or enter q to return to the main menu:

Select the number that corresponds to the device you wish to use. The devices appearing in this menu are taken from */etc/default/archive*; you can add entries or even simplify this menu by editing this file (see “Editing */etc/default/filesys* and */etc/default/archive*” in this chapter).

Note

Take care when selecting the number of the media device. For example, make certain that you don't select “Floppy Drive 1” when you want “Floppy Drive 0.” If you make this error, the backup is aborted and you must start over.



-
5. The following message is displayed:

```
It is important to have plenty of formatted media on hand.
Do you wish to format any media at this time? (y/n)
```

You can format as many volumes as you wish by inserting them into the drive and pressing **RETURN**.

6. Next you see:

```
PERIODIC BACKUP (level 0) - /name FILESYSTEM
```

```
Load the first volume into the drive /dev/name,  
press <RETURN> when you are ready,  
or enter q to return to the main menu:
```

7. Load a volume, tape or disk, into the selected drive, and press **RETURN**. The system displays the current date and the date of the last backup. It displays “the epoch” if there has been no backup. The system then begins to copy files to the drive. If a volume runs out of space, the program displays the message:

```
Please insert new volume, then press <RETURN>
```

8. Remove the first volume, insert a new volume, then press **RETURN**. The program continues to copy files to the new volume. Repeat this step until the program displays the message:

```
DONE
```



6

If you are using floppies, you may need to repeat the last step several times before the backup is complete. You should label each volume as you remove it from the drive. For example, label the first volume “Volume 1,” the second “Volume 2,” and so on.

6.6 Getting a Backup Listing

You can keep a record of the files you have backed by selecting item 3: “List the contents of an archive” from the Filesystem Maintenance Menu. The program copies the names of all files from the backup disks to the temporary file */tmp/backup.list*, or to another file of your choice. This listing is useful when you need to recover a file from a backup, and especially convenient if you wish to keep detailed records of the files copied in each backup.

To get the listing, follow these steps:

1. Log in as **backup**

Δ **sysadmsh** users select: Backups→Create

2. The Filesystem Maintenance menu is displayed:

```
1. Perform a scheduled backup
2. Perform an unscheduled backup
3. List the contents of an archive
4. Restore backed up file(s)
5. Restore an entire filesystem
6. Check backup archive integrity
```

Enter an option or enter q to quit:

3. Enter "3" and press **RETURN**. You are prompted for the name of the file in which to place the listing. Enter **RETURN** if you wish to use the default */tmp/backup.list*. The program prompts you to insert the first backup volume.
4. Load the first volume, then press **RETURN**. The program automatically reads the filenames off the backup volume and places them in the list file.



To print the backup list on a lineprinter, quit **sysadmin** and enter:

```
lp /tmp/backup.list
```

Δ **sysadmsh** users select: Dirs/Files→Print

and press **RETURN**. To save space after printing the file, you should remove it with the **rm(C)** command.

6.7 Restoring Individual Files or Directories From Backups

You can restore individual files or subdirectories from your filesystem backup volumes by invoking **sysadmin** and selecting the fourth item in the Filesystem Maintenance Menu. You will need the complete set of

backup volumes containing the latest version of the file or files you wish to restore. If you are restoring a file that has not been changed recently, use the last level 0 backup.

You must use the "full pathname" of the file or files you wish to restore. This pathname is given in the backup listing. If the files are not on the root filesystem, the name of the filesystem must be omitted from the pathname. For example, to restore the file */u/stellar/data* from your */u* backups, you would enter:

/stellar/data

To restore a file, follow these steps:

1. Log in as **root** and enter:

sysadmin

Δ sysadmsh users select: Backups→Create

and press **RETURN**.

2. When the Filesystem Maintenance Menu appears enter "4" and press **RETURN**. You see:

```
1. Floppy Drive 0 (48dsdd)
2. Floppy Drive 1 (48dsdd)
3. Floppy Drive 0 (96dshd)
4. Floppy Drive 1 (96dshd)
5. Floppy Drive 0 (96dsdd)
6. Floppy Drive 1 (96dsdd)
7. Cartridge Drive (300 ft tape)
8. Cartridge Drive (450 ft tape)
9. Cartridge Drive (600 ft tape)
10. Mini-Cartridge Drive (10MB)
11. Mini-Cartridge Drive (40MB)
12. Other
```

Select an archive device,
or enter q to return to the main menu:

Enter the number that corresponds to the drive used to create the backup originally.

- Next, a message is displayed explaining that you can choose to restore the files to their original location (by providing the name of the top-level directory the filesystem is mounted on), or some other directory. This is followed by a prompt for the directory name:

```
Enter a directory name,  
enter <RETURN> to choose the current directory,  
or enter q to return to the main menu:
```

Note

If you respond with the pathname of the original location, the restored files will overwrite any files by the same names in that location. It is important to be sure that the files on the backup are the desired versions of these files. If you are not absolutely sure that your backup contains the preferred version of the files, you should restore them to a temporary location, such as */tmp*, and compare them with your current files on disk using **diff(C)** or **cmp(C)**.

Enter the name of the directory you want the files restored to.

- Next, you are prompted:

```
List the name of each file or directory to be restored.  
Do not include the destination (filesystem or directory) name,  
but use the rest of the full path name.
```

```
For example, if the destination is /usr, and the full path  
name is /usr/bin/lpr, then enter /bin/lpr.
```

```
Enter a file or directory and <RETURN>,  
enter <RETURN> if the list is complete,  
or enter q to return to the main menu:
```

Enter the full pathname of the files or directories you wish to restore and press **RETURN** to continue the program.



5. Next, you see:

```
RESTORE FILES - /file(s)
Mount desired dump volume:
```

Load volume 1 of the backup set into the drive, then press **RETURN**.

6. The program displays the inode numbers of the files you have given, then prompts for a volume number:

```
/file(s): inode nn
Specify volume #:
```

7. Remove the first volume and replace it with the *last* volume made of the backup set into the drive, enter its number and press **RETURN**. The program searches the volume for the specified files and places copies into the specified locations on your hard disk.
8. The program prompts for volume numbers until all of the files have been found. When each file is found, you see:

```
Extract file filename
```

Continue to feed volumes in reverse order until the first volume made has been loaded and you have returned to the main menu.

6.8 Restoring an Entire Filesystem

Restoring an entire filesystem is a last-resort option, used when a non-root filesystem has become corrupted or unreadable. Do not use this option carelessly; all information currently in the target filesystem will be overwritten. You cannot restore the root filesystem using this utility. If your root filesystem has been corrupted and is not bootable, you can

restore it by referring to “Restoring a Corrupted Root Filesystem” in the “Solving System Problems” chapter of the *XENIX System Administrator’s Guide* and using the Emergency Boot Floppy you created at installation time. If you did not create an Emergency Boot Floppy, you must reinstall XENIX as described in the “Reinstalling and Updating Your System” chapter of the *Installation Guide*.

To restore an entire non-root filesystem, you must first “remake” the filesystem using **divvy**. This will start you with a clear filesystem. Then you can invoke the “Restore entire filesystem” option.

To restore a filesystem, follow this procedure:

1. Log in as the super-user (**root**) and enter one of the following commands:

If you are restoring a filesystem on the primary disk (/dev/hd00):

```
divvy -b 1 -c 1 -p 0
```

If you are restoring a filesystem on the secondary disk (/dev/hd10):

```
divvy -b 1 -c 1 -p 1
```

- You see a table similar to the following, plus the main **divvy** menu:

Name	New File System?	#	First Block	Last Block
root	no, exists	0	0	13754
swap	no, exists	1	13755	15135
u	no, exists	2	15136	25135
	no	3	—	—
	no	4	—	—
	no	5	—	—
recover	no, exists	6	25136	25145
dl057all	no	7	0	25546

x blocks for divisions, y blocks reserved for the system

n[ame] Name or rename a division.
 c[reate] Create a new filesystem on this division.
 p[revent] Prevent a new filesystem from being created...
 s[tart] Start a division on a different block.
 e[nd] End a division on a different block.
 r[estore] Restore the original partition table.

Please enter your choice or 'q' to quit:

- Enter "c" to recreate a filesystem. You are then prompted for a division number as displayed in column three:

which division? (0 through 6) --

Enter the number corresponding to the filesystem you wish to recreate.

Note

You should take extreme care when selecting the filesystem to recreate. However, after quitting out of **divvy**, you can undo any mistakes by selecting "e[xit]" as described in the next step.

- The **divvy** menu is displayed again, but with “yes” in the “New File System?” column. Enter “q” to quit.
- Next, you are given a final chance to undo your changes before leaving **divvy**:

```
i[install]   Install the division set-up shown
r[return]   Return to the previous menu
e[exit]     Exit without installing a division table
```

Please enter your choice:

If you made a mistake, enter “e”, otherwise enter “i” to use your changes. You can then quit out of **divvy** entirely. When you quit, your filesystem is rebuilt. You see the message:

```
Making Filesystems
```

- Next, while still logged in as root, enter the command:

sysadmin

△ **sysadmsh** users select: Backups→Restore



When the Filesystem Maintenance Menu is displayed, select option “5”. You are warned:

```
WARNING: RESTORING A LEVEL 0 BACKUP WILL OVERWRITE YOUR FILESYSTEM.
```

Even if a filesystem has been damaged, it may contain valuable information. It is very important to make sure that the set of backups you restore from has not also been damaged before restoring directly onto the damaged filesystem.

This option should be used to restore a level 0 backup onto a clear, or newly created, filesystem. The most recent level 9 backup with a later creation date than the level 0 should then be restored onto it.

Do you wish to continue? (y/n)

Respond “y” if you are certain that this is what you wish to do.

7. Next you are asked to select the filesystem that you wish to restore:

```
1. / - the root filesystem
2. /u -
3. Other
```

Select a filesystem to restore
or enter q to return to the main menu:

If you select "Other", you are prompted for the device name of the filesystem that you wish to restore.

8. Next you see the archive menu. Select the medium on which your filesystem is backed up, for example, tape or floppy.
9. You are prompted to load the first volume of the backup into the selected drive:

```
RESTORE FILESYSTEM - /name
```

Load the first backup volume into drive /dev/devicename.
Press <RETURN> when you are ready,
or enter q to return to the main menu:

Start with the last complete (level 0) backup, loading each volume in order as prompted.

10. You are first given another chance to stop:

```
Last chance before scribbling on /dev/name.
```

If you wish to continue, press **RETURN**.

11. The restoration process may take some time. When the restore phase of the operation is complete, you see:

```
End of backup
The restore of /name has been successful.
The filesystem will be checked to insure integrity.
```

Next, **sysadmin** runs a check on the filesystem using **fsck(ADM)**. You see a series of messages like this:

```
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Free List
***** FILE SYSTEM WAS MODIFIED *****
```

12. When the filesystem check is complete, you see:

```
You must remount /dev/name when ready to use the filesystem.
```

13. Now that the restore of your level 0 volumes is complete, you are returned to the Filesystem Maintenance Menu. Repeat steps 6-12 until each of your higher level backups (1,8 and 9) have been restored that were done between the last level 0 and the date when your filesystem was damaged. Be sure and restore them in the order they were done, or you will overwrite recent versions of files with older ones.

Your filesystem is now completely restored. You need to mount the filesystem before you can use it. To mount it, enter the following command, substituting *name* for the name of your filesystem:



```
mount /dev/name /name
```

6.9 Editing /etc/default/filesys and /etc/default/archive

The files */etc/default/filesys* and */etc/default/archive* are used by **sysadmin** to create the filesystem and archive device menus. Even though each menu provides the option “Other” so that you can use filesystems and devices not described, you should keep these default files up to date as your system changes. These files are also used by other programs, and should be maintained as specified for these programs.

/etc/default/archive

The */etc/default/archive* file contains a complete set of devices supported by XENIX. Each device, filesystem or drive, in these files is represented by a one line entry which consists of "name=value" pairs, separated by spaces or tabs. For example, the following is a possible entry in */etc/default/archive*:

```
cdev=/dev/rfd048ds9 desc="Floppy Drive 0 (48dsdd)" \  
blocking=18 size=360 format="format -f /dev/rfd048ds9"
```

The value part of "name=value" pairs "desc="Floppy Drive 0 (48dsdd)"" and "format="format -f /dev/rfd048ds9"" contain spaces, therefore they must be surrounded by quotes in order to be interpreted correctly.

If your system does not use certain drives included in */etc/default/archive*, or if you add an entry to either file, and then later decide that you don't need it any longer, rather than deleting the entry, you can place a pound sign (#) at the beginning of the first line for that device, and it will be treated as a comment and ignored. Later, if you need the entry again, you can delete the #.

Entries commented out will not appear in the list of media devices displayed in the "Restore backup file(s)" option of the Filesystem Maintenance Menu.



/etc/default/filesys

The minimum necessary information about a filesystem is:

- A character device name, (cdev=), or a block device name, (bdev=).
- You may include a description, (desc=), which appears in the Filesystem Maintenance Menu.
- The mount directory, (mountdir=), which is also used by the **mnt(C)** utility, in addition to **sysadmin**. The name *rcmount*, in the example below is used exclusively by */etc/rc*.

```
bdev=/dev/root cdev=/dev/rroot mountdir=/ \  
desc="The Root Filesystem" rcmount=no mount=no
```

```
bdev=/dev/u cdev=/dev/ru mountdir=/u rcmount=yes \  
fsckflags=-y desc="The User Filesystem"
```


6.10 An Explanation of Backup Levels

The most straightforward and dependable way to ensure the safety of data is to back up everything on a filesystem at one time. However, filesystems tend to be quite large (from 10 to 300 MB), and may take hours to backup. The concept of backup levels (or incremental backups) addresses this problem. The general idea of an incremental backup is to back up only those files that have changed since a previous backup. This can significantly reduce the size and duration of the backup. Consider the following scheme:

Monthly	complete backup
Weekly	everything newer than last week
Daily	everything newer than yesterday

This means that at the end of every month, the entire filesystem is backed-up. Each week, the files that have changed since last week are backed-up, and each day, any files that have changed since yesterday. If at some point a filesystem is damaged, you would simply restore the last full (monthly) backup, the last weekly backup, and any daily backups that happened just prior to the accident. Thus it is always possible to reconstruct a filesystem from a series of backups.

While this is a simple method to understand, the implementation using incremental backup levels is not.

6.10.1 Principles of Incremental Backup Levels

To make the business of backing up files more efficient, XENIX uses a progressive series of levels, each of which is based on the last occurrence of a lower level backup. The XENIX backup facility provides up to ten different levels of backups, giving the system administrator tremendous flexibility in organizing backups.

Level	Files Saved
0	all files on the filesystem
1	files changed since last level 0 backup
2	files changed since last level 1 backup
3	files changed since last level 2 backup
..	-----
9	files changed since last level 8 backup

The full ten levels would be used to accommodate computers with massive filesystems; average systems will use only a few levels. The levels



serve to subdivide a backup into manageable units. It is important to realize that each backup level creates backups based on the previous (next lowest) level backup. This means that the order of the backups is not significant, but the level number is.

For example, let's assume that the following backups were done for a week:

Day	Level	Files Backed Up
Mon	0	All files on filesystem
Tue	5	All files changed since Monday
Wed	2	All files changed since Monday
Thu	7	All files changed since Tuesday
Fri	5	All files changed since Wednesday

This example is illogical, but serves to demonstrate how the levels work. Remember that each of the backups saves the files changed since the next lower level backup, and that level 0 is the lowest. Therefore, the level 5 on Friday backs up all files changed since the next lowest number, level 2, on Wednesday. The level 5 on Monday saves only those files that have changed since the day before, since the only previous lower level backup is a 0. If all the backup levels except Monday were level 5, each would still backup all files that changed since the level 0 on Monday.

6.10.2 How the Default Schedule Works

The default **schedule** file provided with your distribution uses only four levels, and is optimized for use on systems under moderate use (7-10 terminals, 8-10 users):

#		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
#	Filesystem	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F
	/dev/rroot	0	x	9	x	9	8	x	9	x	9	1	x	9	x	9	8	x	9	x	9
	/dev/ru	9	0	9	9	9	9	8	9	9	9	9	1	9	9	9	9	8	9	9	9

Figure 6-6: The Default Schedule

The /u Filesystem

Filesystem */dev/ru* is a heavily-used resource. Some level of backup is performed every day. This scheme is designed to minimize resources while maximizing safety; if one or more of the backups for that week is lost or goes bad, there is sufficient redundancy to minimize any loss of data.

According to the default schedule, a full (level 0) backup of */dev/ru* occurs at the beginning of the month. (Because a level 0 is done on the root filesystem on Monday, the level 0 for */u* is done on Tuesday.) Wednesday, a level 9 backup saves just those files on */dev/ru* which have changed since the level 0 backup. By the end of the week far fewer floppies or tapes are used than the number needed for full backups each day. Time is substantially reduced as well. If it is necessary to restore the filesystem to the last recorded state, you would restore the last level 0 backup, followed by each of the lower-level backups that have been done since.

Note that each Tuesday, a lower level backup (0, 1 or 8) occurs that saves everything since the beginning of the month and causes each of the level 9 that follow it to be based on that week. This way the level 9 backups don't become too large and redundant.

The root Filesystem

The root filesystem contains the operating system and other system files. It changes less frequently, so it is not backed up every day. Each Monday, a lower level backup is done, and level 9 backups are done twice per week. Just as with the */u* filesystem, the level 9 backups are restricted to cover only those files that have changed during that week.

6

6.10.3 How Backups are Used to Restore a Filesystem

Now, let's assume you have a hardware failure that ruins the information on the hard disk. Assume it happens on the last Thursday of the month, just before the backup was to be done that evening. You fix the hardware problem and reinstall your system, but how do you restore your backups? Restore the last occurrence of each backup level, in ascending order:

- level 0 (done on the first Tuesday of the month)
- level 1 (done on the third Tuesday)

XENIX System Administrator's Guide

- level 8 (done on the fourth Tuesday)
- level 9 (done on Wednesday evening)

You wouldn't need to restore the level 8 that was done on the second Tuesday, because the level 1 that followed it covered the same files. The only information that is missing is what was changed during the day on Thursday, just before the crash. This is the primary reason for backups; recovery should be straightforward and with a minimum of loss.

Chapter 7

Adding Device Drivers

with the Link Kit

- 7.1 Introduction 7-1
- 7.2 Device Drivers 7-1
 - 7.2.1 Installing Device Drivers 7-2
 - 7.2.2 Installing Preconfigured Drivers 7-3
 - 7.2.3 Installing Older Drivers and Drivers Without Shell Scripts 7-4
 - 7.2.4 Troubleshooting 7-8
 - 7.2.5 Creating Special Device Files 7-8
- 7.3 Freeing Kernel Space for Drivers 7-9
- 7.4 Testing and Installing the New Kernel 7-10
 - 7.4.1 Booting the New Kernel 7-10
 - 7.4.2 Creating A New /xenix 7-11
 - 7.4.3 Removing the Link Kit 7-11



7.1 Introduction

This chapter explains how to add device drivers to the XENIX kernel, a process known as “linking.” To change any component of the XENIX kernel it is necessary to use the Link Kit to relink the kernel. The Link Kit consists of a set of kernel components in the form of relocatable object modules plus various programs and shell scripts used to link the components together.

The most common use for the Link Kit is to add device drivers to the system. A device driver is the software interface between a peripheral device and the operating system. Each device that can be used with XENIX must have a device driver. New drivers are generally supplied when adding a peripheral device to the system; they must be configured into XENIX before the device will function.

The Link Kit is also necessary to create new device drivers, and is used in conjunction with other tools that are included with the XENIX Development System. The device driver material in this chapter is intended for those who want to install an existing device driver. Driver writers should read the “Writing Device Drivers” chapter of the *C User's Guide*. Example device drivers are in the “Sample Device Drivers” chapter of the *C User's Guide*.

7.2 Device Drivers

A device driver is a set of routines that communicates with a hardware device, and provides a means by which XENIX can control the device in order to perform Input/Output (I/O) operations.

A device driver is usually supplied as a single software module. Installing this software into the kernel is as important as the actual hardware installation. It must be completed before the device can be used. A driver is usually accompanied by an auxiliary program or shell script that helps to form the links between driver and kernel.

To install a new device driver:

- Install the hardware device on the system according to the manufacturer's instructions.
- Boot the system and enter system maintenance mode. All the operations described as part of the installation process are carried out in this mode

- Make sure the Link Kit is installed. If it is not already installed, install it using the **custom(ADM)** command.
- If the kernel has already been modified, copy the configuration files back from where they were previously stored. For example, enter:

```
cd /usr/sys/conf
cp ../io/master ../io/xenixconf ../io/link_xenix ../io/driver.o
```

where *driver.o* is the name of the driver installed earlier, if any.


7.2.1 Installing Device Drivers

The exact instructions for installing a new device driver are different for each type of device. This section contains example commands that may be slightly different than the actual commands. Read the specific installation instructions that are provided with the device driver software.

After the Link Kit is installed and the instructions read, the next step depends on how much of the work has already been done by the driver's vendors.

Many software vendors provide automatic driver installation utilities compatible with the standard System V installation utilities. If so, insert the vendor's floppy in the floppy drive and enter:

custom

 Select the option to add a supported product, and follow the instructions that appear on the screen. **custom** should run any System V compatible, automatic installation software provided with the driver. This installs the device driver software and links a version of the kernel that contains the new device driver. After **custom** completes, the next step is usually to test the newly created kernel. See the device driver documentation for details.

If no mention is made of **custom** in the documentation for the individual driver, use the following procedure.

Move to the directory containing the Link Kit and copy the necessary files by entering:

```
cd /usr/sys/conf
cp master master.old
cp xenixconf xenixconf.old
```

Some older installation sets contain their own versions of **master** and **xenixconf**. These files record the system configuration. If the installation floppy contains its own **master** and **xenixconf** files, an out-of-date and possibly obsolete kernel will be generated.

Insert the floppy containing the driver into the floppy drive and enter the following to extract the contents of the installation floppy:

```
tar xvf /dev/install
```

Examine the names of the extracted files. If there are no files named **master**, **xenixconf**, **c.o**, or **c.c** on the driver installation floppy, the driver is *preconfigured*. Proceed to “Installing Preconfigured Drivers” in this chapter. Otherwise, proceed to “Installing Older Drivers and Drivers Without Configuration Shell Scripts” to determine the commands necessary to configure the driver.

7.2.2 Installing Preconfigured Drivers

The driver installation floppy may come with a shell script that edits the link command line to include the new driver. If such a script is present, run it by entering:

```
./scriptname
```

where *scriptname* is the name of the script. Most scripts also create all necessary device nodes; if this is the case, proceed to “Booting the New Kernel.” If your script does not create the proper nodes, you must create them with the **mknod**(ADM) command. For more information on making nodes, please refer to the **mknod**(ADM) manual page.



If no such script is present, edit the file *link_xenix* to include the names of all object files provided. The object files are the files on the distribution media whose names end in “.o”, as in *tape.o*. Add the names of any new modules to the **ld** command line, just before the pairs of arguments of the form “-l lib_xxxx”.

Enter:

```
./link_xenix
```

Linking can take as long as 10 minutes. Once a new XENIX kernel has been created, proceed to “Creating Special Device Files.”

7.2.3 Installing Older Drivers and Drivers Without Shell Scripts

Any driver configuration shell script that neither installs a preconfigured driver nor calls **configure** dates from before the current release of XENIX. Many older drivers will work fine with the current XENIX release, but their shell scripts are no longer usable. If this is the case, the script must be ignored and the procedure is the same as for a driver unaccompanied by a configuration shell script.

Note

There is no warranty, expressed or implied, that any driver working with any earlier release will remain compatible with any subsequent kernel. Most drivers will continue to be compatible with new releases of XENIX. The following procedure will allow you to configure in an earlier driver. However, there are many reasons why an earlier driver may no longer function, few of them directly determinable.

Outdated drivers may simply malfunction when attempting to access their peripherals, or they may cause a system crash or even more insidious malfunctions. For this reason, make backup copies of all important files before attempting to reconfigure an older driver, and test drivers only in single user (maintenance mode).

1. Repeat the following procedure for each device driver. Enter:

```
rm c.o c*.o space.o space*.o
```

Ignore any error messages of the form “c.o non-existent” or “space.o non-existent.”

2. If files named *master* and *xenixconf* were extracted when the **tar** command was entered, copy the old *master* and *xenixconf* files back to their original names. This will overwrite the versions of *master* and *xenixconf* that came with the driver. For example, use the commands:

```
cp master.old master
cp xenixconf.old xenixconf
```

3. Enter the following to obtain the *Major Device Number* (write it down for later use):

```
./configure -j NEXTMAJOR
```

4. The driver module is the remaining file or group of files from the installation media whose names end in “.o”. Enter:

```
./routines module1.o module2.o ...
```

where *module1.o module2.o ...* are the list of the driver modules provided. The list is most likely one module long, but if there is more than one, list them all.

This command can take as long as 5 minutes. Write down the names produced. Most of these names are either configurable driver routines or driver priority levels. Some names may be spurious.

5. Driver priority levels have names consisting of the string *spl* followed by a number between 0 and 7. If there are any strings beginning with *spl* present, write down the largest such number under the heading *Interrupt Priority Level*. Then cross all *spl* routines off the list. For example, if the string *spl6* is produced and it is the highest priority level produced, write it down under the heading *Interrupt Priority Level*.
6. Configurable driver routines all have a common prefix, such as *sio*. Each prefix is followed by one of a small group of suffixes: *open*, *close*, *read*, *write*, *ioctl*, *strategy*, *halt*, *poll*, *intr*, *init*, *tab*, *_ty*, or

stream. If there are files that do not fit this pattern, cross them out. For example, running **routines** on the *sio.o* driver reveals a long list of routines that begin with *sio*, and a single routine, *ttinit*. In this case you would cross out *ttinit* because it doesn't begin with *sio*. There are a few other routines in the *sio* driver that would also be crossed out, such as *siopinit*, because of the extra "p." *sio* is an extreme case: most drivers will not have spurious routine names scattered throughout the relevant ones.

7. If any routine ends with *strategy* or *tab*, the peripheral is a *block* device. If any routine ends with *read*, *write*, or *ioctl*, the device is a *character* device. A peripheral may be both a *block* and a *character* device. If none of these routines are present, consider the peripheral a *character* device.
8. If there is a routine containing the name *intr*, refer to the hardware manual to figure out which vector or vectors the device is capable of interrupting. To get a list of the vectors that are currently in use, enter:

```
./vectorsinuse
```

A few drivers are written to allow vector sharing, but it is better to give each device a unique vector whenever possible. Associate the peripheral with an appropriate vector or vectors. Write down the numbers chosen for the "Vectors".

9. The **configure** command has the following syntax and must be entered on a single line, without pressing RETURN until the entire command has been entered:

```
./configure -b -c -m Major_Device_Number -v Vector_or_Vector_List  
-a List_Of_Driver_Routines -l Interrupt_Priority_Level
```

The options have the following definitions and restrictions:

- b** Use if configuring a "block" device.
- c** Use if configuring a "character" device.
- m** Should be followed by the *Major_Device_Number* determined earlier.
- a** Should be followed by the list of driver routines determined by running **routines** and crossing out the extraneous entries.

Adding Device Drivers with the Link Kit

- v Use only if the device has an *intr* routine; should be followed by the list of vectors determined earlier.
- l Use only if *spl* routines appeared when **routines** was run earlier, followed by the *Interrupt_Priority_Level*.

For example, if you configure the serial I/O driver, use the following command:

```
./configure -c -m 5 -v 3 4 27 28 -a sioopen sioclose sioread  
sioclock sioioctl siointr siopoll sioinit -l 7
```

The ramdisk driver is a simpler example; if it is not present, you can add it with the command:

```
./configure -b -m 31 -a ramopen ramclose ramstrategy ramtab
```

configure will produce new *c.o* and *space.o* files containing updated configuration information.

Note

If **routines** produces a long list containing a number of different prefixes, each with a healthy complement of configuration suffixes, this driver package contains not one driver but a driver suite made up of several drivers. Treat each prefix as an individual driver, and run **configure** once for each prefix.

-
10. After **configure** has been run once for each driver, edit the file *link_xenix* to include the names of all object files provided. The object files are the files on the distribution media whose names end in “.o”, as in *tape.o*. Add the names of any new modules to the **ld** command line, just before the pairs of arguments of the form “-l lib_xxxx”. Enter:

```
./link_xenix
```

Note that linking can take a while. Once a new XENIX kernel has been linked, proceed to “Creating Special Device Files.”



7.2.4 Troubleshooting

If the following **ld** error message appears on 286 machines:

```
Group ``DGROUP`` larger than 64Kbytes
```

Reduce the size of some of the other kernel data structures to compensate for the extra room that the new driver is using. See "Freeing Kernel Space For Drivers" in this chapter for detailed instructions.

For hand-configured drivers: if the new kernel links without error, but on boot-up gets to the letter D and no farther (Ln on 386 machines, where *n* is a single digit), it is possible that an *init* routine produced when **routines** was run was never meant to be given on the **configure** command line. After rebooting the old XENIX kernel, enter the following commands:

```
cd /usr/sys/conf
./configure -d XXinit -m Major_Device_Number -b -c
./link_xenix
```

where *Major_Device_Number* is the value determined earlier, and *XXinit* is the actual name of the initialization routine. Run **link_xenix** again, copy the kernel produced to the root directory, reboot, and try again.

7.2.5 Creating Special Device Files

For programs to gain access to the newly installed devices, they must also exist as files within the filesystem. These files are called *special files* and are usually located in the */dev* directory. Once again, the specific installation instructions supplied with the device will give the precise details of the name to be used for the *special file* and the other parameters associated with it. To create a *special file*, use the **mknod** command. Supply the name of the special file, its type (which can be either "b" for a *block* device or "c" for a *character* device) and the *major* and *minor* device numbers associated with the device. For example, change directories to */dev* and enter a command similar to one of these:

```
/etc/mknod hcd0 b 1 0
/etc/mknod rhcd0 c 1 0
/etc/mknod hqp c 7 0
```

Note the convention for setting up disk device names. A digit may be appended to the mnemonic to indicate the drive number. The "raw" device, or *character special* device, name has an "r" prefix.

The major device number of the device can be found in the “master” file. Find a line in this file for an appropriate device. For example, a tape driver might be called “tape” at the beginning of the line, and “td” somewhat further down the same line. The name of the driver should also correspond to the name of the object module. For example, *tape.o* should be called “tape” inside the master file.

Next, find the columns at the top of the file marked “bmaj” and “cmaj”. Search down these two columns until a line appears that describes the driver. Write down the block major device number (bmaj), and the character major device number (cmaj). If either bmaj or cmaj is “0”, do not create any block or character nodes, respectively. Otherwise, these entries are the major device numbers for the driver.

7.3 Freeing Kernel Space for Drivers

If a driver is too large to be linked into the kernel, it may be possible to free sufficient kernel space by reducing certain resource allocations. Refer to “Reallocating Kernel Resources with *configure*” in the “Tuning System Performance” chapter.

This type of resource fine-tuning is similar to reconfiguring for performance. A reasonable first attempt would be to halve the allocations of each resource in *configure*’s “Message Queue”, “Shared Data”, and “Semaphore” categories, except for the semaphore values SEMVMX, SEMAEM, and SEMOPM, which are not strictly resources and cause no space to be allocated. Halving the allocation of the other resources in these categories will free approximately 2.2K.

Note



No space will be freed by decreasing external buffers (NBUF), files (NFILE), inodes (NINODE) or multiscreens (NSCRN), as each of these resources is allocated out of user memory, not the precious near data that drivers must occupy.

After *configure* completes, the kernel is ready to link. Enter:

```
./link_xenix
```

Note that linking can take a while under XENIX-286. Reboot the system as described below to test the new kernel. Kernel space is generally not a

problem under XENIX-386. Kernels larger than 570K cannot boot under -286. If this is the case, reboot your previous kernel and unlink unused portions to reduce kernel size to less than 570K. You can use the **size(C)** utility to display the current size of your kernel.

7.4 Testing and Installing the New Kernel

7.4.1 Booting the New Kernel

Test the new kernel before installing it as */xenix*. To do so, enter the following two command lines:

```
cp /usr/sys/conf/xenix /xenix.new  
/etc/reboot
```

The system now reboots. A boot prompt appears:

```
Boot  
:
```

If you press RETURN, or do not enter anything for a time, the default operating system image */xenix* is loaded and started. In order to test the newly installed device drivers, enter the name of the new kernel at the boot prompt:

```
xenix.new
```

7

and press RETURN. The system is now running with the “new” kernel. Test the various devices (especially any that have been added).

Be aware that when an alternate kernel is used, **ps(C)** does not work correctly unless the **-n** flag and the pathname of the alternate XENIX kernel is specified. For example:

```
ps -n /xenix.new
```

Whenever a different kernel is booted, remove */usr/adm/messages* before switching to multi-user mode.

Note

Do not install *xenix* on the hard disk as */xenix* until it is fully tested.

7.4.2 Creating A New */xenix*

When the kernel is satisfactory, install the new kernel on the hard disk. Enter the following:

```
cd /usr/sys/conf
./hdinstall
```

hdinstall(ADM) moves the “old” */xenix* to a file called */xenix.old* and copies */usr/sys/conf/xenix* to become */xenix*.

7.4.3 Removing the Link Kit

Once the kernel is made, tested, and installed as */xenix*, you can use **custom** to remove the Link Kit to save disk space. Before you do this, it is important to save the *master*, *xenixconf*, *driver.o*, and *link_xenix* files. */usr/sys/io* is one suitable place for them. These files are records of system changes. Unless these files are readily available, any further system configuration performed after removing and reinstalling the Link Kit will negate the work performed to reconfigure the system. Type:

```
cd /usr/sys/conf
cp master xenixconf link_xenix driver.o ./io
```

Where *driver.o* is the driver or list of drivers.

At the beginning of this chapter you were instructed to copy any previously created backup files into */usr/sys/conf* if they existed. Now they exist. If these files are protected, an accurate picture of the system configuration can be maintained over any reconfigurations to come.





Chapter 8

Tuning System Performance

- 8.1 Introduction 8-1
 - 8.1.1 Examples of Specialized Resource Allocation 8-2
- 8.2 Reallocating Kernel Resources with `configure` 8-2
 - 8.2.1 Using the `configure` Command Line 8-3
- 8.3 Reconfiguring Because of Persistent Error Messages 8-4
- 8.4 Reconfiguring for Performance 8-5
 - 8.4.1 Deallocating Unused Resources 8-5
 - 8.4.2 Improving Disk Utilization 8-5
 - 8.4.3 Filesystem Organization 8-8
- 8.5 Defining Efficient System Usage Patterns 8-10
 - 8.5.1 `ps` 8-10
 - 8.5.2 User `$PATH` Variables 8-10
- 8.6 Using `vmstat` to Diagnose System Inefficiency 8-11
 - 8.6.1 Memory Usage: Buffers Versus Pages 8-11
 - 8.6.2 The `vmstat` Display 8-12
 - 8.6.3 Checking Buffer and Page Cache Usage 8-13
 - 8.6.4 Checking CPU Usage 8-13
- 8.7 Summary of Tunable Parameters 8-14
 - 8.7.1 Disk Buffers 8-18
 - 8.7.2 Character Buffers 8-19
 - 8.7.3 Files, Inodes, and Filesystems 8-19
 - 8.7.4 Processes, Memory Management & Swapping 8-21
 - 8.7.5 Clock 8-22
 - 8.7.6 `MultiScreens` 8-22
 - 8.7.7 Message Queues 8-23
 - 8.7.8 Semaphores 8-24
 - 8.7.9 Shared Data 8-25
 - 8.7.10 System Name 8-25
 - 8.7.11 Streams Data 8-26
 - 8.7.12 Event Queues and Devices 8-29

8.7.13 Hardware Dependent Parameters 8-29

8.1 Introduction

The Link Kit contains the **configure**(ADM) utility, which is used to alter a number of parameters that affect system performance. This chapter explains how to change these parameters to suit the needs of your system. In addition, general procedures are included that can improve resource usage and system performance.

Your XENIX system is optimized for use with a variety of hardware configurations and as a platform for many applications. The kernel, which lies at the heart of the operating system, controls a number of resources that are constantly being used, released and and recycled. These resources include:

buffers	A cache of in-memory storage units used to hold recently used data. (Buffers increase efficiency by keeping this data on hand and decrease reading from the disk.)
table entries	A space in any of many tables that the kernel uses to keep track of the current tasks, resources and events.
other parameters	Other definable values govern special resources (such as the number of multiscreens available or quantity of semaphores).

The use of these resources is defined by certain limits that can be decreased or extended, sometimes at the expense of other resources.

Performance tuning is an activity that may need your attention when you first set up your XENIX system. When you bring the system up for the first time, the system is automatically set to a basic configuration that is satisfactory for most applications. This configuration, however, cannot take into account the usage patterns and the behavior of your particular applications. For this reason, the structure of the system allows you to reconfigure it to enhance the performance for your particular application over that of the standard configuration.

There are several reasons for reallocating system resources:

- You install additional hardware memory and thus have greater memory resources to allocate.
- Persistent error messages are displayed indicating that certain resources are used up, such as inodes or table entries.

- The system response time is consistently slow, indicating that other resources are too constrained for the system to operate efficiently (as when too little hardware memory is installed.)
- Resource usage needs to be tailored to meet the needs of a particular application.

In addition, it is important to determine which resources are being wasted or have become inefficiently distributed due to the natural tendency for order to become chaos.

8.1.1 Examples of Specialized Resource Allocation

Specialized applications often require the reallocation of key system resources for optimum performance. For example, users with large databases may find that they need to lock more files simultaneously than the current allocation of file locks will permit. Users who have no need for specialized XENIX features such as message handling may find that they can get a slight performance boost by deallocating those features.

Deciding how to best optimize the use of these resources is known as performance or kernel tuning. Each resource or limit is represented by a separate kernel parameter. The actual values are altered using the **configure** utility.

8.2 Reallocating Kernel Resources with configure

The **configure** utility is an easy-to-use menu-driven program that presents each resource and prompts for modification. To set the allocation of the appropriate resources, relink the kernel by invoking the shell script *link_xenix*, copy the kernel to the root directory, reboot, and test the new kernel.

To change any kernel parameter, do the following:

1. If the Link Kit is not already installed, install it using the **custom(ADM)** command.
2. After making certain the Link Kit is installed, enter the following commands:

```
cd /usr/sys/conf
```

```
./configure
```

3. The **configure** menu is displayed:

- ```
1. Disk Buffers
2. Character Buffers
3. Files, Inodes, and Filesystems
4. Processes, Memory Management & Swapping
5. Clock
6. MultiScreens
7. Message Queues
8. Semaphores
9. Shared Data
10. System Name
11. Streams Data
12. Event Queues and Devices
13. Hardware Dependent Parameters
```

```
Select a parameter category to reconfigure by
typing a number from 1 to 13, or type 'q' to quit:
```

Choose a category by entering the number preceding it. The resources in that category will be displayed, one by one, each with its current value. Enter a new value for the resource, or to retain the current value, simply press RETURN. After all the resources in the category have been displayed, **configure** will return to the category menu prompt. Choose another category to reconfigure or exit **configure** by entering **q**.

4. After you have finished changing parameters, you must link them into a new kernel. Enter the following command:

```
./link_xenix
```

This assembles each of the kernel modules into a new kernel, which must now be installed. Follow the instructions in the “Testing and Installing the New Kernel” section of the “Adding Device Drivers with the Link Kit” chapter.

### 8.2.1 Using the **configure** Command Line

**configure** also has a command-line interface suitable for use by application developers. For instance, a database developer who finds that 70 files rather than 50 files need to be locked simultaneously may provide a shell

script to perform the reconfiguration. To find the current value of any configurable resource using the command-line interface, enter:

```
./configure -y RESOURCE
```

Where *RESOURCE* is the name of the tunable parameter (in uppercase). To change the value of any resource from the command line, enter:

```
./configure RESOURCE=value
```

This interface is in addition to the interactive one; the same resources are configurable from both interfaces.

The sections that follow describe scenarios for reconfiguring the kernel resources.

## 8.3 Reconfiguring Because of Persistent Error Messages

The kernel should not be reconfigured because a kernel error message was received once, or even a couple of times, but when a single message persists. First try to increase a resource by a small amount, and if the problem persists, increase it by 50 or even 100 percent of its original value. If the problem is still not solved, more detailed research will be required to locate the exact program and sequence that causes the error.

### System Messages

Inode Table Overflow

Increase NINODE in the "Files, Inodes, and Filesystems" category. NFILE and NINODE are usually set equal, but NINODE may be less in environments where many links are common. You can check these values with **pstat(C)**.

no file

Increase NFILE in the "Files, Inodes, and Filesystems" category. NFILE and NINODE are usually set equal, but NINODE may be less in environments where many links are common. You can check these values with **pstat(C)**.



PANIC: Timeout table overflow

Increase NCALL in the "Clock" category.

out of text

Increase NTEXT in the "Processes, Memory Management & Swapping" category. (Applies only to 80286-based machines.)



no more processes

Increase NPROC in the “Processes, Memory Management & Swapping” category.

map overflow (*n*) shutdown and reboot

Increase CMAPSIZ and SMAPSIZ parameters in the “Processes, Memory Management and Swapping” category. (Applies only to 80286-based machines.)

### 8.4 Reconfiguring for Performance

The system is configured such that greatest quantities of kernel resources are assigned to the most common tasks such as reading and writing from the disk, but performance of the more specialized features (such as inter-process communication) has not been ignored. This balance can be shifted to conform to individual requirements.

#### 8.4.1 Deallocating Unused Resources

In order to save precious kernel space, it is possible to deallocate resources that your system does not use. By deallocating all semaphores, about 1.5K can be freed. By deallocating all message queue structures, about 3K can be freed. No standard utilities use either of these constructs, but they should only be entirely deallocated if no applications are present that might use these constructs. By deallocating all shared data segments, approximately 1.2K can be freed. Since shared data is used by applications such as spreadsheets and by the Cmerge compiler in some memory models, it is usually inappropriate to deallocate shared data entirely. In addition, if your system does not have programs that use file locking, you can decrease this resource (NFLOCKS) accordingly.

#### 8.4.2 Improving Disk Utilization

Disk input/output may cause a bottleneck in system performance. There are three steps in tuning the disk subsystem for better utilization.

- Choosing the proper number of buffers.
- Setting the sticky-bit on selected programs.
- Organizing the filesystems to minimize disk activity.



## Choosing Buffer Size

When the **configure** utility “Disk Buffer” category is selected, the buffer resources NBUF, NSABUF (XENIX-286 only), NHBUF and MAXBUF will be displayed in turn. (Buffer allocation differs in XENIX-286; refer to “Buffer Allocation on 80286-based Machines.”)

Use the number of buffers (NBUF and NHBUF) given in Table 8-1 as a starting point. These values come close to optimum for most system workloads.

The NBUF parameter controls the number of buffers in the system buffer cache used to reduce the need to access the disks. These buffers hold recently used data on the chance that it will be needed again. NHBUF specifies the number of hashing buckets in the buffer cache. The more buffers, the greater chance that needed data can be found in the buffers without the system having to do a time-consuming disk read. If the value for NBUF is left null, the system calculates the values shown in Table 8-1. The value for NHBUF is a power of 2 that is roughly one-quarter the value of NBUF.

When the number of buffers, NBUF, is nonzero, its value is the actual number of disk buffers. When NBUF is zero, the system will autoconfigure buffers based on the amount of memory available. To find out how many buffers have been autoconfigured, examine the value of “i/o bufs” displayed during system boot.

Increasing NBUF and NHBUF, up to a point, improves system performance. A system with two megabytes of memory can typically devote roughly 250K bytes of memory to buffers (250 buffers at approximately 1K bytes each) while a system with 4 megabytes of memory can devote 600K bytes of memory to buffers (more if there are few users). However, if too many buffers are allocated, there may not be enough memory space for efficient operation of user processes, and the amount of swapping done by the system will increase. The swapping activity usually costs more in system efficiency than is gained by having a large amount of buffer space.



After your XENIX system has run for a day or so, you will want to check for excessive swapping activity. If such activity is found, reduce the number of buffers (NBUF and NHBUF) or increase your system memory. (See the “Using vmstat to Diagnose System Inefficiency” section for more information.)

### Buffer Allocation on 80286-based Machines

On 80286-based machines, there is a distinction made between near and far kernel data. NSABUF determines the number of near disk buffers and NBUF determines the number of far disk buffers. (This distinction is meaningless on 80386-based machines and the parameter NSABUF is not used.) The quantity of System Addressable Buffers (NSABUF), represents a tradeoff between buffers and other kernel resources.

A small amount of kernel resource space is available for additional device drivers, but it can be allocated to NSABUF, if desired. Beyond this, NSABUF should only be increased if other kernel resources are not being used.

The basic scheme is to deallocate any unused resources, set MAXBUF and NBUF to equal, large numbers, and after finding an appropriate tradeoff between user memory and external disk buffers, use any remaining near data space space for increasing NSABUF and the few extra buffer headers that the additional buffers will require.

Under XENIX-286, if combined allocated resources consume more than 64K of memory, the `ld` error message:

```
Group "DGROUP" larger than 64Kbytes
```

appears when the shell script `link_xenix` is run. This message does not apply to SCO XENIX-386. If you see this message, you should reduce some of the configurable kernel parameters discussed in this chapter.

### Setting the “Sticky-Bit” (80286 Machines Only)

Setting the sticky bit can reduce the disk traffic of a select group of commands. The term “sticky” refers to making a program stay, or “stick” in memory. The text segments of selected sticky commands are kept in memory or contiguously in swap space, even when the process terminates.



## Note

The sticky-bit is only useful on 80286-based machines. With 80386-based machines, recently used pages (with freshly initialized data) are held in the page cache, so there is no need to set the sticky bit. If the bit is set anyway, it is ignored.

---

The sticky bit is normally useful for very frequently used programs like **ls**, **sh**, and **vi**. Any program used often enough can have the sticky bit set for a significant performance boost. The sticky bit can be set by entering the following command:

```
chmod u+t filename
```

## 8.4.3 Filesystem Organization

This section describes several actions that can be taken to reduce the overhead of file access. As filesystems are used, the blocks of individual member files tend to become physically scattered around the disk(s) and I/O becomes less efficient. This scattering yields poor ordering of blocks with files and poor directory structure.

### Organization of Filesystem Free List

Filesystems are set up to allocate free blocks in a manner that allows the files to be read or written with efficiency. A free list array is created when a filesystem is created with **mkfs(ADM)**. The free list is set up with the rotational gap specified by **mkfs** options. The difference between successive block numbers in the free list is the rotational gap. For example, a file created on a system with a rotational gap of 10 may consist of blocks 510, 520, and 530. When the file is read, I/O requests are sent to the disk drive to read blocks 510, 520, and 530. As soon as the drive finishes reading block 510 and has started to process the second request, block 520 will be moving over the read/write head just as the drive is ready to read that request. This method makes for efficient I/O operation.

However, as you start changing files (changing size or removing), the efficiency starts to decrease. When several files are being created at once, they will be contending for blocks from the free list. Some of the blocks allocated to the files will be out of sequence. As you can see, the free list also becomes scattered about the disk as blocks are allocated and freed.

### Directory Organization

Directory organization also affects input/output performance. The problems show up when files are removed by users. When a file is removed from a directory, the i-node number is nulled out. This leaves an unused slot for that i-node; over time the number of empty slots may become quite large. If you have a directory with 100 files in it and you remove the first 99 files, the directory still contains the 99 empty slots, at 16 bytes per slot, preceding the active slot. In effect, unless a directory is reorganized on the disk, it will retain the largest size it has ever achieved.

### Restoring Good Filesystem Organization

There is no automatic way to solve these problems; however, you can manually rearrange the filesystem. Here are a variety of ways to do this. Note that in all cases the filesystem(s) must be unmounted.

1. To reorganize the free list, run **fsck(ADM)** using the **-s** option.
2. To reorganize particular directory structures, use **cpio(C) -pdm** to copy them to a temporary location, remove the original structure, then use **cpio -pdm** to copy them back to their original location. Use the following command line:

```
find sourcedir -print | cpio -pdm destdir
```

*sourcedir* is the name of the original directory; *destdir* is the directory you are moving the structure to. After confirming that the entire structure has been copied over, remove the entire original directory. Finally, swap the source and destination directory names in the command line above, and execute the command again. After removing the superfluous directory, your original directory structure is reorganized.

3. To reorganize an entire filesystem, use the **sysadmin** utility to perform an unscheduled level 0 backup of the filesystem. (This is described in detail in the “Backing Up Filesystems” chapter in this Guide.) When the backup is complete, follow the instructions for restoring an entire filesystem.
4. If you have more than one disk, balance heavily-used filesystems across the disks.



## 8.5 Defining Efficient System Usage Patterns

After the kernel and the system activities are tuned, and the file systems organized, the next step for improving system performance is to perform some housekeeping activities and to check whether prime time load can be reduced. The person responsible for administering the system should check for:

- less important jobs interfering with more important jobs
- unnecessary activities being carried out
- scheduling of selected jobs for when the system is not so busy
- the efficiency of user-defined features, such as *.profile* and \$PATH

### 8.5.1 ps

The **ps(C)** command is used to obtain information about active processes. The command gives a "snapshot" picture of what is going on, which is useful when you are trying to identify what processes are loading the system. Things will probably change by the time the output appears; however, the entries that you should be interested in are **TIME** (minutes and seconds of CPU time used by processes) and **STIME** (time when process first started).

When you spot a "runaway" process (one that uses progressively more system resources over a period of time while you are monitoring it), you should check with the owner. It is possible that such a process should be stopped immediately via the **kill(C)** command. When you have a real runaway, it continues to eat up system resources until everything grinds to a halt. For a process that proves "unkillable," rebooting is the only way to stop it.

When you spot processes that take a very long time to execute you should consider using **cron(C)** to execute the job during off-hours.

8

### 8.5.2 User \$PATH Variables

\$PATH is searched upon each command execution. Before displaying "not found," the system must search every directory in \$PATH. These searches require both processor and disk time. If there is a disk or processor bottleneck, changes here can help performance.

Some things that you should check for in user \$PATH variables are:

- *Path Efficiency*

\$PATH is read left to right, so the most likely places to find the command should be first in the path (**/bin** and **/usr/bin**). Make sure that a directory is not searched more than once for a command.

- *Convenience and Human Factors*

Users may prefer to have the current directory listed first in the path (**:/bin**).

- *Path Length*

In general, \$PATH should have as few entries as possible.

- *Large Directory Searches*

Searches of large directories should be avoided if possible. Put any large directories at the end of \$PATH.

## 8.6 Using vmstat to Diagnose System Inefficiency

The **vmstat** utility can be a useful tool for examining and tuning system performance. It cannot provide definitive answers on resource tuning issues, but can provide some insight into the internal workings of the system and help diagnose problems related to poor memory and CPU usage.

---

### Note

The **vmstat** utility is only applicable to 80386-based machines and is not available for the 80286.



### 8.6.1 Memory Usage: Buffers Versus Pages

Specifying the number of system disk buffers (either explicitly using **configure** or implicitly using the defaults), effectively divides the available memory between two pools: the disk buffer pool and the page pool. The page pool contains the programs being run and cached copies of

recently-used program pages. If the page pool is much too small for the load imposed on the system, the system will be constantly swapping pages in and out just to keep up with the current processes.

If the page pool is only slightly undersized, the effects will be seen not in swapping overhead but in reduced cache performance when running the same programs repeatedly. This means that sufficient pages are available to effectively handle current processes, but there are none to spare for keeping recently-used pages in memory for potential access savings.

By using **vmstat**, you can determine how many programs have been swapped in and out during a given interval. If excessive swapping is evident, you can redistribute memory allocation to increase the page pool. We also recommend adding as much RAM as practical; swapping is decreased and performance is improved.

8.6.2 The vmstat Display

**vmstat** is used to take a series of snapshots of how the system is performing, including:

- A summary of the number of processes in various states,
- Paging activity,
- system activity,
- CPU cycle consumption.

The **vmstat** display looks like this:

| procs |    |   |    | paging |    |    |     |     |     |     |     | system |     |     |    | cpu |    |    |    |    |
|-------|----|---|----|--------|----|----|-----|-----|-----|-----|-----|--------|-----|-----|----|-----|----|----|----|----|
| r     | b  | w | si | so     | ch | cm | ffr | swr | sww | rec | shf | shc    | cpy | pf  | in | sy  | cs | us | su | id |
| 2     | 67 | 0 | 0  | 0      | 0  | 0  | 0   | 0   | 0   | 0   | 0   | 0      | 0   | 155 | 68 | 198 | 26 | 72 | 20 | 8  |



For the purposes of this procedure, the only categories you are interested in are as follows:

- si**            Number of processes swapped in.
- so**            Number of processes swapped out.



|           |                                                                                                                    |
|-----------|--------------------------------------------------------------------------------------------------------------------|
| <b>ch</b> | Number of page cache hits; the number of pages that were re-used rather than retrieved from disk.                  |
| <b>cm</b> | Number of page cache misses; essentially the number of times the system had to retrieve a page from the hard disk. |
| <b>id</b> | The percentage of time the CPU spent doing nothing; idle time.                                                     |

### 8.6.3 Checking Buffer and Page Cache Usage

To determine whether you have enough memory in the page pool for the page caching mechanism to work efficiently, do the following:

1. Run a small set of programs over and over in addition to your usual process and user load.
2. Start the **vmstat** command running at fifteen second intervals by entering the following:

```
vmstat 15
```

Watch the columns “ch” (page cache hits) and “cm” (page cache misses) for a few minutes. If you see more misses than hits, you may find that a larger page pool will help performance. (However, an initially poor page cache hit-to-miss ratio is to be expected when running programs for the first time.) If the page pool is too small, the usual solution is to add more memory, but there may be application mixes for which shrinking the buffer cache will achieve the same result.

### 8.6.4 Checking CPU Usage

The “id” (percentage idle) column may also provide some insight into system performance. This figure is normally between 40 and 100 percent, even with a large number of active users. When this figure falls consistently below 30%, the chief competition for resources does not involve memory at all; the critical resource is raw processor power. (Run the **ps(C)** command to make certain that the excessive CPU usage is not due to a runaway process that is stealing every spare CPU cycle.)

If you are running a large number of users, it may help to switching to smart serial boards if you are using more common dumb cards. Smart cards take some of the burden off the CPU rather than adding to the amount of work it has to do.



In addition, you should examine `/usr/spool/crontab` to see if jobs are queued up for peak periods that might better be run at times when the system is idle. Use the `ps` command to determine what processes are heavily loading the system. Encourage users to run large, non-interactive commands (such as `nroff(CT)` or `troff(CT)`) at off-peak hours. You may also want to run such commands with a low priority by using the `nice(C)` or `batch(C)` commands.

### 8.7 Summary of Tunable Parameters

Tunable system parameters are used to set various table sizes and system thresholds to handle the expected system load. Caution should be used when changing these variables since such changes can directly affect system performance. For the most part, the default tunable parameter values are acceptable for most configurations and applications. If your application has special performance needs, you may have to experiment with different combinations of parameter values to find an optimal set.

Table 8-1 shows the recommended tunable parameter values for a system equipped with different amounts of hardware memory (RAM) installed.

The following notes apply to the Tunable Parameters Table 8-1:

- The value of a few parameters are calculated each time a new kernel (`/xenix`) is generated, unless the value is manually overridden (see note below).
- The default value and the size in bytes for each entry are shown in the table.
- A dash (—) is used in the size information to indicate parameters that set flags in the kernel. Parameters that set flags do not affect the size of the kernel when their values are changed; only the values of the specific flags are changed.
- A dagger (†) indicates a parameter that applies only to 80386-based machines.
- An asterisk (\*) indicates a parameter that applies only to 80286-based machines.



### *Note*

Overriding the calculated value causes the parameter to be set to the new value each time a new kernel (`/xenix`) is generated. When calculated parameter values are overridden, however, any subsequent changes in the hardware configuration (adding new memory for example) require a change of the value that was manually set. This will allow you to optimize the performance of the new configuration.

---

# XENIX System Administrator's Guide

| Parameter | RAM Installed |        |          |           | Default Value | Size per Entry in Bytes |
|-----------|---------------|--------|----------|-----------|---------------|-------------------------|
|           | 1 MEG         | 2 MEGS | 3-5 MEGS | 6-15 MEGS |               |                         |
| NDISK     | —             | —      | —        | —         | CALC.         | —                       |
| NBUF      | 100           | 250    | 400      | 600       | CALC.         | 1024                    |
| NSABUF*   | —             | —      | —        | —         | 0             | —                       |
| NPBUF     | 4             | 16     | 40       | 60        | 10            | 52                      |
| NHBUF     | 32            | 64     | 128      | 256       | 128           | 12                      |
| MAXBUF    | —             | —      | —        | —         | 600           | —                       |
| NCLIST    | 100           | 100    | 160      | 200       | 100           | 72                      |
| NEMAP     | —             | —      | —        | —         | 10            | —                       |
| NSXT      | —             | —      | —        | —         | 1             | —                       |
| NINODE    | 100           | 225    | 300      | 400       | 100           | 68                      |
| NFILE     | 100           | 225    | 300      | 400       | 100           | 12                      |
| NMOUNT    | 8             | 8      | 8        | 8         | 8             | 36                      |
| NFLOCKS   | —             | —      | —        | —         | 50            | —                       |
| CMASK     | —             | —      | —        | —         | 0             | —                       |
| NPROC     | 60            | 70     | 80       | 100       | 60            | 168                     |
| MAXUPRC   | 25            | 25     | 30       | 30        | 30            | —                       |
| MEMLIM†   | —             | —      | —        | —         | 100           | —                       |
| SWPLIM†   | —             | —      | —        | —         | 30            | —                       |
| CMAPIZ*   | —             | —      | —        | —         | CALC.         | —                       |
| SMAPSIZ*  | —             | —      | —        | —         | CALC.         | —                       |
| NTEXT*    | —             | —      | —        | —         | 1             | —                       |
| NCALL     | —             | —      | —        | —         | 70            | —                       |
| TIMEZONE  | —             | —      | —        | —         | 480           | —                       |
| DSTFLAG   | —             | —      | —        | —         | 1             | —                       |
| NSCRN     | —             | —      | —        | —         | 0             | —                       |
| SCRNMEM   | —             | —      | —        | —         | CALC.         | —                       |
| MSGMAP    | 513           | 513    | 513      | 513       | 513           | 8                       |
| MSGMAX    | 8192          | 8192   | 8192     | 8192      | 8192          | —                       |
| MSGMNB    | 8192          | 8192   | 8192     | 8192      | 8192          | —                       |
| MSGMNI    | 10            | 10     | 10       | 10        | 10            | 53                      |
| MSGTQL    | 60            | 60     | 60       | 60        | 60            | 12                      |
| MSGSSZ    | 8             | 8      | 8        | 8         | 8             | 1024                    |
| MSGSEG    | 1024          | 1024   | 1024     | 1024      | CALC.         | 8                       |
| SEMMAP    | 21            | 21     | 21       | 21        | 21            | 8                       |
| SEMMNI    | 10            | 10     | 10       | 10        | 10            | 32                      |
| SEMMNU    | 20            | 20     | 20       | 20        | 20            | 8X(SEMUME+2)            |
| SEMMSL    | 10            | 10     | 10       | 10        | 10            | —                       |

**Table 8-1: Configurable Kernel Parameters**

## Tuning System Performance

| Parameter  | RAM Installed |        |          |           | Default Value | Size per Entry in Bytes |
|------------|---------------|--------|----------|-----------|---------------|-------------------------|
|            | 1 MEG         | 2 MEGS | 3-5 MEGS | 6-15 MEGS |               |                         |
| SEMOPM     | 5             | 5      | 5        | 5         | 5             | 8                       |
| SEMUME     | 5             | 5      | 5        | 5         | 5             | 8XSEMMNU                |
| SEMVMX     | 32766         | 32766  | 32766    | 32766     | 32766         | —                       |
| SEMAEM     | 16384         | 16384  | 16384    | 16384     | 16384         | —                       |
| SEMMNS     | 40            | 40     | 40       | 40        | 40            | 8                       |
| NSDSEGS†   | 25            | 25     | 25       | 25        | 25            | —                       |
| NSDSLOTS†  | 3             | 3      | 3        | 3         | 3             | —                       |
| SHMMIN†    | 1             | 1      | 1        | 1         | 1             | —                       |
| SHMMNI†    | 25            | 25     | 25       | 25        | 25            | 52                      |
| SHMSEG†    | 6             | 6      | 6        | 6         | 6             | 12XNPROC                |
| SHMALL†    | 4096          | 4096   | 4096     | 4096      | 4096          | —                       |
| NODE       | —             | —      | —        | —         | —             | —                       |
| NQUEUE     | —             | 256    | 384      | 384       | 128           | 36                      |
| NSTREAM    | —             | 32     | 48       | 48        | 32            | 52                      |
| NBLK8192   | —             | 0      | 0        | 0         | 0             | 4142                    |
| NBLK4096   | —             | 0      | 0        | 0         | 5             | 4142                    |
| NBLK2048   | —             | 20     | 40       | 40        | 5             | 2094                    |
| NBLK1024   | —             | 12     | 32       | 32        | 10            | 1070                    |
| NBLK512    | —             | 8      | 18       | 18        | 10            | 558                     |
| NBLK256    | —             | 16     | 48       | 48        | 5             | 302                     |
| NBLK128    | —             | 64     | 128      | 128       | 5             | 174                     |
| NBLK64     | —             | 256    | 256      | 256       | 64            | 110                     |
| NBLK32     | —             | 128    | 128      | 128       | 128           | 110                     |
| NBLK16     | —             | 128    | 256      | 256       | 0             | 61                      |
| NBLK4      | —             | 128    | 128      | 128       | 128           | 50                      |
| NMUXLINK   | —             | 32     | 48       | 48        | 32            | 12                      |
| NSTREVENT  | —             | 256    | 256      | 256       | 64            | 12                      |
| NSTRPUSH   | —             | 9      | 9        | 9         | 16            | —                       |
| MAXSEPGCNT | —             | 1      | 2        | 2         | 1             | 2048                    |
| STRMSGSZ   | —             | 4096   | 4096     | 4096      | 4096          | —                       |
| STRCTLSZ   | —             | 1024   | 1024     | 1024      | 512           | —                       |
| STRLOFRAC  | —             | 80     | 80       | 80        | 80            | —                       |
| STRMEDFRAC | —             | 90     | 90       | 90        | 90            | —                       |
| EVQUEUES   | 1             | 3      | 6        | 40        | 8             | —                       |
| EVDEVS     | 2             | 6      | 12       | 36        | —             | 16                      |
| EVDEVSPERQ | 2             | 3      | 3        | 32        | 3             | —                       |
| DMAEXCL    | —             | —      | —        | —         | —             | —                       |
| KBTYPE     | —             | —      | —        | —         | —             | —                       |



**Table 8-1: Configurable Kernel Parameters (continued)**

The following is a complete list of configurable parameters, their purpose, and suggested tuning values.

### 8.7.1 Disk Buffers

- NDISK**                   The number of floppy drives attached to the system. This is set at boot time.
- NBUF**                    Specifies how many system buffers to allocate. The XENIX system buffers form a data cache. The data cache is a memory array containing disk file information. Improvement in the hit rate of this cache increases with the number of buffers. Cache hits reduce the number of disk accesses and thus improve overall performance. The entries are normally in the range of 100 to 600. Each buffer contains 1076 bytes. Hash buffers (NHBUF) should be increased along with system buffers (NBUF) for optimal performance.
- NSABUF**                The number of buffer cache buffers in near kernel data on 286 kernels. Irrelevant on 386 machines since all data is near.
- NPBUF**                Specifies how many physical input/output buffers to allocate. Each entry contains 52 bytes. The default value is 8.
- NHBUF**                Specifies how many "hash buckets" to allocate. These are used to search for a buffer given a device number and block number rather than a linear search through the entire list of buffers. **This value must be a power of 2.** Each entry contains 12 bytes. The NHBUF value should be chosen so that the value NBUF divided by NHBUF is approximately equal to 4. While the value of NBUF is normally calculated by the system, that is not true for NHBUF.
- MAXBUF**               Maximum possible number of buffer cache buffers. This is the number of buffer description headers in the kernel. Fewer than this number of buffers may actually be autoconfigured by the kernel at boot time, depending on how much core is present. If NBUF is non-zero, then exactly NBUF buffers will

be configured, and there is no reason for MAXBUF to be larger than NBUF. If NBUF is 0, the kernel will configure at most MAXBUF buffers automatically.

### 8.7.2 Character Buffers

#### **NCLIST**

Clists, or character lists, are small buffers used to read from and write to serial devices. If NCLIST is insufficient, serial I/O performance may suffer. Each buffer contains up to 64 bytes. The buffers are dynamically linked to form input and output queues for the terminal lines and other slow speed devices. The average number of buffers needed per terminal is in the range of 5 to 10. No additional performance can possibly be achieved after there are 16 clists per character device, and most systems need far fewer. Each entry (buffer space plus header) contains 72 bytes. When the CLISTS are full, input and output characters dealing with terminals are lost, although echoing continues on the screen.

#### **NEMAP**

The number of 8-bit channel maps used for character set mapping. Each corresponds to one possible alternate character set.

#### **NSXT**

The number of shell-layer sessions.

### 8.7.3 Files, Inodes, and Filesystems

#### **NINODE**

Specifies how many inode table entries to allocate. Each table entry represents an in-core inode that is an active file. For example, an active file might be a current directory, an open file, or a mount point. The file control structure is modified when changing this variable. The number of entries used depends on the number of opened files. The entries are normally in the range of 100 to 400. The value for NINODE pertains directly to the NFILE value. Although these two resources are usually set equal, they sometimes diverge, for example:

The number of open files is greater than the number



of open inodes whenever two processes have the same file open. The number of open inodes is greater than the number of open files while a file is being opened.

When the inode table overflows, the following warning message is output on the system console:

```
inode table overflow
```

## NFILE

Specifies how many open file table entries to allocate. Each entry represents an open file. The entry is normally in the range of 100 to 400. Each entry contains 12 bytes. The NFILE entry relates directly to the NINODE entry. The NFILE control structure operates in the same manner as the NINODE structure. When the file table overflows, the following warning message is output on the system console.

```
file table overflow
```

As a reminder, this parameter does not affect the number of open files per process.

## NMOUNT

Determines the maximum number of filesystems that can be mounted at one time. The root filesystem counts as a mounted filesystem in this calculation. If you configure this parameter, make sure to allow for an extra filesystem so that you can mount a filesystem floppy. When full, the **mount(S)** system call returns the error EBUSY. Since the mount table is searched linearly, this value should be as low as possible.

## NFLOCKS

The number of files that can be locked at once. If you are not using an application that uses file locking, (none of the standard XENIX utilities do) this resource can be partially deallocated to save space. The default value is 50. Each entry contains 28 bytes.

## CMASK

The mask used for file creation.





### 8.7.4 Processes, Memory Management & Swapping

#### **NPROC**

The number of processes that can be active anywhere in the system. A process attempting to fork when there are already NPROC processes active will receive the error EAGAIN (see intro(S)). This error also generates the “no more processes” screen message. If any significant change is made to NPROC, CMAPSIZ and SMAPSIZ (286 only) should also be changed (see below). The swapper is always the first process and **/etc/init** is always the second. The number of entries depends on the number of terminal lines available and the number of processes spawned by each user. The average number of processes per user is in the range of 2 to 5 (also see MAXUP, default value 30). The NPROC entry is in the range of 50 to 200.

#### **MAXUPRC**

The number of processes that a single user can run simultaneously. A process attempting to fork when the user already has MAXUPRC processes active will receive the error EAGAIN. The entry is normally in the range of 15 to 30. This value should not exceed the value of NPROC (NPROC should be at least 10% more than MAXUP). This value is per user identification number, not per terminal. For example, if 12 people are logged in on the same user identification, the default limit would be reached very quickly.

#### **MEMLIM**

A process may occupy up to this percent of user memory, plus the swap area it can occupy (which is constrained by SWPLIM). (This parameter is only valid on 80386-based machines.)

#### **SWPLIM**

A process may occupy up to this percent of swap area, plus the memory area it can occupy (which is constrained by MEMLIM). (This parameter is only valid on 80386-based machines.)

#### **CMAPSIZ**

Table used to hold list of program segments in memory. (This parameter is only valid on 80286-based machines.)

#### **SMAPSIZ**

Table used to hold lists of program segments being swapped. While CMAPSIZ and SMAPSIZ rarely



need to be changed themselves, they should vary with NPROC, each being  $NPROC * 2$  by default. (This parameter is only valid on 80286-based machines.)

**NTEXT** Maximum text segments (systemwide) (This parameter is only valid on 80286-based machines.)

## 8.7.5 Clock

**NCALL** Specifies how many call-out table entries to allocate. Each entry represents a function to be invoked at a later time by the clock handler portion of the kernel. This value must be greater than 2 and is normally in the range of 10 to 70. The default value is 60. Each entry contains 16 bytes.

An example of callout table entry usage: Software drivers may use call entries to check hardware device status. When the call-out table overflows, the system crashes and outputs the following message on the system console:

```
PANIC: Timeout table overflow
```

**TIMEZONE** Number of minutes west of Greenwich Mean Time.

**DSTFLAG** This flag defines daylight savings time.

## 8.7.6 MultiScreens

**NSCRN** The number of multiscreens that can be used on the console. The maximum is 12.

**SCRNMEM** The number of 1024 byte blocks for console screen saves.



## 8.7.7 Message Queues

The following tunable parameters are associated with inter-process communication messages.


- MSGMAP** Specifies the size of the control map used to manage message segments. Default value is 513. Each entry contains 8 bytes.
- MSGMAX** Specifies the maximum size of a message in bytes. The default value is 8192. The maximum size is 64 kilobytes -1 byte.
- MSGMNB** Specifies the maximum length of a message queue. The default value is 8192.
- MSGMNI** Specifies the maximum number of message queues system-wide (id structure). The default value is 10.
- MSGTQL** Specifies the number of message headers in the system and, thus, the number of outstanding messages. The default value is 60. Each entry contains 12 bytes.
- MSGSSZ** Specifies the size, in bytes, of a message segment. Messages consist of a contiguous set of message segments large enough to fit the text. The default value is 8. The value of MSGSSZ times the value of MSGSEG must be less than or equal to 131,072 bytes (128 kilobytes).
- MSGSEG** Specifies the number of message segments in the system. segments during system initialization. The default value is 1024. MSGSEG and MSGSSZ are multiplied to determine the number of bytes of memory to be allocated for messages. The value of  $MSGSEG * MSGSSZ$  is the number printed out in the XENIX boot message (e.g. "msg bufs = 8K"). Note that there is a flag `IPC_NOWAIT` that can be passed into many of the `msg()` system calls. If this flag is passed, the system calls will fail immediately if there is no space for a message. If this flag is not passed, then the system calls will sleep until there is room for the message. The value of MSGSSZ times the value of MSGSEG must be less than or equal to 131,072 bytes (128 kilobytes). If MSGSEG is zero, its value is set relative to the



amount of memory in the system at boot time.  $\text{MSGSEG} * \text{MSGSSZ}$  should be no more than  $\text{MSGTQL} * \text{MSGMAX}$ , but may be smaller to conserve user memory.

## 8.7.8 Semaphores

The following tunable parameters are associated with inter-process communication semaphores.

- |                                                                                                |                                                                                                                                                                                                       |
|------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>SEMMAP</b>                                                                                  | Specifies the size of the control map used to manage semaphore sets. The default value is 21. Each entry contains 8 bytes.                                                                            |
| <b>SEMMNI</b>                                                                                  | Specifies the number of semaphore identifiers in the kernel. This is the number of unique semaphore sets that can be active at any given time. The default value is 10. Each entry contains 32 bytes. |
| <b>SEMMNS</b>                                                                                  | Specifies the number of semaphores in the system. The default value is 40. Each entry contains 8 bytes.                                                                                               |
| <b>SEMMNU</b>                                                                                  | Specifies the number of undo structures in the system. The default value is 30. The size is equal to $8 * (\text{SEMUME} + 2)$ bytes.                                                                 |
| <b>SEMMSL</b>                                                                                  | Specifies the maximum number of semaphores per semaphore identifier. The default value is 10.                                                                                                         |
| <b>SEMOPM</b>                                                                                  | Specifies the maximum number of semaphore operations that can be executed per <b>semop(S)</b> system call. The default value is 5. Each entry contains 8 bytes.                                       |
|  <b>SEMUME</b> | Specifies the maximum number of undo entries per undo structure. The default value is 5. The size is equal to $8 * (\text{SEMMNU})$ bytes.                                                            |
| <b>SEVMX</b>                                                                                   | Specifies the maximum value a semaphore can have. The default value is 32767. The default value is the maximum value for this parameter.                                                              |
| <b>SEMAEM</b>                                                                                  | Specifies the adjustment on exit for maximum value, alias <b>semadj</b> . This value is used when a semaphore value becomes greater than or equal to                                                  |

the absolute value of **semop(S)**, unless the program has set its own value. The default value is 16384. The default value is the maximum value for this parameter.

### 8.7.9 Shared Data

The following tunable parameters are associated with inter-process communication shared memory. (These parameters are only valid on 80386-based machines.)

|                 |                                                                                                                                |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>NSDSEGS</b>  | The maximum number of shared memory segments allowed in the system.                                                            |
| <b>NSDSLOTS</b> | $NSDSLOTS * NSDSEGS$ is the maximum number of simultaneous attaches to shared memory segments for the entire system.           |
| <b>SHMMAX</b>   | Specifies the maximum shared memory segment size. The default value is 131072.                                                 |
| <b>SHMMIN</b>   | Specifies the minimum shared memory segment size. The default value is 1.                                                      |
| <b>SHMMNI</b>   | Specifies the maximum number of shared memory identifiers system wide. The default value is 100. Each entry contains 52 bytes. |
| <b>SHMSEG</b>   | Specifies the number of attached shared memory segments per process. The default value is 6. The maximum value is 15.          |
| <b>SHMALL</b>   | Specifies the maximum number of in-use shared memory text segments. The default value is 512.                                  |

### 8.7.10 System Name

|             |                                        |
|-------------|----------------------------------------|
| <b>NODE</b> | Specifies the node name of the system. |
|-------------|----------------------------------------|



## 8.7.11 Streams Data


The following tunable parameters are associated with Streams processing. The values should be left at 0 unless the STREAMS package has been installed and you're planning on using STREAMS. Exception: STREVENT and NMUXLINK should be set to 1.

---

### Note

STREAMS support requires installation of the XENIX STREAMS Runtime package; it is not included with the XENIX Operating System distribution.

---

- |                                                                                                  |                                                                                                                                                                                                                                                                                                                                                                                                               |
|--------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>NQUEUE</b>                                                                                    | The number of STREAMS queues to be configured. Queues are always allocated in pairs, so this number should be even. A minimal Stream contains four queues (two for the Stream head, two for the driver). Each module pushed on a Stream requires an additional two queues. A typical configuration value is $8 * NSTREAM$ .                                                                                   |
| <b>NSTREAM</b>                                                                                   | The number of "Stream-head" (stdata) structures to be configured. One is needed for each Stream opened, including both Streams currently open from user processes and Streams linked under multiplexers. The recommended configuration value is highly application-dependent, but a value of 32-40 usually suffices on an 80386-based computer for running a single transport provider with moderate traffic. |
|  <b>NMUXLINK</b> | The maximum number of multiplexer links to be configured. One link structure is required for each active multiplexor link (STREAMS I_LINK ioctl). This number is application dependent; the default allocation equal to the number of Streams (NSTREAM) guarantees availability of links.                                                                                                                     |
| <b>NBLK<sub>n</sub></b>                                                                          | The number of STREAMS data blocks and buffers to be allocated for each size class. Message block headers are also allocated based on                                                                                                                                                                                                                                                                          |

these numbers: the number of message blocks is 1.25 times the total of all data block allocations. This provides a message block for each data block, plus some extras for duplicating messages (kernel functions **dupb()**, **dupmsg()**). The optimal configuration depends on both the amount of primary memory available and the intended application.

### **NSTREVENT**

The initial number of Stream event cells to be configured. Stream event cells are used for recording process-specific information in the **poll(S)** system call. They are also used in the implementation of the **STREAMS I\_SETSIG ioctl** and in the kernel **bufcall()** mechanism. A rough minimum value to configure would be the expected number of processes to be simultaneously using **poll(S)** times the expected number of Streams being polled per process, plus the expected number of processes expected to be using **STREAMS** concurrently. The default is 256. Note that this number is not necessarily a hard upper limit on the number of event cells that will be available on the system (see **MAXSEPGCNT**).

### **NSTRPUSH**

The maximum number of modules that may be pushed onto a Stream. This is used to prevent an errant user process from consuming all of the available queues on a single Stream.

### **MAXSEPGCNT**

The number of additional pages of memory that can be dynamically allocated for event cells. If this value is 0, only the allocation defined by **NSTREVENT** is available for use. If the value is not 0 and if the kernel runs out of event cells, it will under some circumstances attempt to allocate an extra page of memory from which new event cells can be created. **MAXSEPGCNT** places a limit on the number of pages that can be allocated for this purpose. On a 80386-based computer, each new page can provide 340 event cells. Once a page has been allocated for event cells, however, it cannot be recovered later for use elsewhere. It is recommended that the **NSTREVENT** value be set to accommodate most load conditions, and that



MAXSEPGCNT be set to 1 to handle exceptional load cases should they arise.

## STRMSGSZ

The maximum allowable size of the data portion of any STREAMS message. This should usually be set just large enough to accommodate the maximum packet size restrictions of the configured STREAMS modules. If it is larger than necessary, a single **write(S)** or **putmsg(S)** can consume an inordinate number of message blocks. The recommend value of 4096 is sufficient for existing applications.

## STRCTLSZ

The maximum allowable size of the control portion of any STREAMS message. The control portion of a **putmsg(S)** message is not subject to the constraints of the min/max packet size, so the value entered here is the only way of providing a limit for the control part of a message. The recommended value of 1024 is more than sufficient for existing applications.

## STRLOFRAC

The percentage of data blocks of a given class at which low-priority block allocation requests are automatically failed. For example, if STRLOFRAC is 80 and there are 48 256-byte blocks, a low-priority allocation request will fail when more than 38 256-byte blocks are already allocated. The parameter is used to help prevent deadlock situations by starving out low-priority activity. The recommended value of 80 works well for current applications. STRLOFRAC must always be in the range  $0 \leq \text{STRLOFRAC} \leq \text{STRMEDFRAC}$ .

## STRMEDFRAC

The percentage cutoff at which medium priority block allocations are failed (see STRLOFRAC discussion above). The recommended value of 90 works well for current applications. STRMEDFRAC must always be in the range  $\text{STRLOFRAC} \leq \text{STRMEDFRAC} \leq 100$ .



### *Note*

There is no cutoff fraction for high-priority allocation requests; it is effectively 100.

---

### 8.6.12 Event Queues and Devices

|                   |                                                                |
|-------------------|----------------------------------------------------------------|
| <b>EVQUEUEES</b>  | Maximum number of open event queues system-wide.               |
| <b>EVDEVS</b>     | Maximum number of devices attached to event queues systemwide. |
| <b>EVDEVSPERQ</b> | Maximum devices per event queue.                               |

### 8.6.13 Hardware Dependent Parameters

|                |                                                                        |
|----------------|------------------------------------------------------------------------|
| <b>DMAEXCL</b> | This is set to 1 if only 1 DMA channel is usable at once, 0 otherwise. |
| <b>KBTYPE</b>  | This is set to XT for XT-type keyboards and AT for AT-type keyboards.  |





# Chapter 9

## Using DOS and XENIX

### On the Same Disk

---

- 9.1 Introduction 9-1
- 9.2 Partitioning the Hard Disk Using **fdisk** 9-1
- 9.3 Installing XENIX on a DOS System 9-4
- 9.4 Using XENIX and DOS With Two Hard Disks 9-6
- 9.5 Removing an Operating System From the Hard Disk 9-7
- 9.6 DOS Accessing Utilities 9-7
- 9.7 XENIX and DOS On Non-Standard Disks 9-8



## 9.1 Introduction

Many users received the MS-DOS, or other closely compatible DOS, operating system with their computer. This chapter explains how you can still use DOS utilities, files, and applications after you install the XENIX operating system. You can even access DOS files and directories from XENIX. XENIX provides this facility so that you do not need to throw away your investment in DOS software, or buy another computer just to run XENIX.

Several programs make this coexistence possible. The **dos(C)** utilities allow access to DOS files on diskettes or on the DOS partition on the hard disk. These utilities are discussed later in this chapter. The utility which partitions the disk is called **fdisk(ADM)** and is available in DOS and XENIX versions. The next section explains how to use **fdisk** to run DOS and XENIX on the same hard disk. Another section discusses installing XENIX on the hard disk along with DOS. There is also a section explaining various booting configurations, for users who mostly use XENIX and for users who mostly use DOS.

## 9.2 Partitioning the Hard Disk Using **fdisk**

Each version of **fdisk** is documented in the respective operating system's manual. **fdisk(ADM)** is found in this guide. and, unless otherwise noted, this chapter refers to the XENIX **fdisk**.

**fdisk** is interactive, and uses a menu to display your options. Here is the main **fdisk** menu:

- ```
1. Display Partition Table
2. Use Entire Disk For XENIX
3. Create XENIX Partition
4. Activate Partition
5. Delete XENIX Partition
```

```
Enter your choice or 'q' to quit:
```

The **fdisk** utility allows you to set up separate areas (partitions) on your hard disk for your operating system. The hard disk is divided into *tracks*. The number of tracks depends upon the size of the hard disk.



A *partition* consists of a group of tracks. One hard disk may contain up to four partitions. Each partition can have a different operating system and associated directories and files.

The **fdisk** command allows you to specify a disk partition as “active”. This means that when you turn on (boot) your computer, the operating system installed in the active partition will start running. The XENIX partition must be active when you intend to use the XENIX operating system.

The **fdisk** command allows you to specify the number of tracks assigned to each partition. The number of available tracks will vary according to the size of your hard disk. We recommend using at least a 30 megabyte hard disk to run XENIX. The size of the XENIX partition also depends on the number of software packages you want to install. Refer to the **custom(ADM)** manual page for information on how to install and remove packages from the three XENIX distribution Systems. You will generally need at least six megabytes for your XENIX partition. You can install the XENIX Operating System package in this space, and have space for user files.

The **fdisk** command allows you to specify where the partition begins. **fdisk** will not allow you to construct overlapping partitions. You do not need to install XENIX in the first partition.

You should always start your DOS partition at the beginning of the disk, starting at cylinder 1, not cylinder 0. Because DOS writes the boot block on cylinder 0 very close to the end of the Masterboot block, starting your DOS partition on cylinder 0 can cause the DOS partition to become inaccessible after installation.

If you install XENIX on the same disk after DOS, start the XENIX partition at the beginning of the next cylinder on the disk. To find the beginning of the next cylinder, note the ending track number of your DOS partition and start the XENIX partition on the next track number that is a multiple of the number of heads on your hard disk. For example, if you have five heads on your hard disk and your DOS partition ends at track 103, start your XENIX partition at track 105.

When you are running XENIX, the device name of the partition running XENIX is */dev/hd0a*.

9

One option of **fdisk** tabulates the current state of the partitions (the Display Partition Table option). This option lists, for each partition, whether the partition is active, the first track, the last track, the number of tracks used, and the associated operating system.

If you enter the Display Partition Table option and press **RETURN** to see the partition table, the result will be similar to this:

Current Hard Disk Drive: /dev/hd00

Partition	Status	Type	Start	End	Size
1	Inactive	DOS	005	398	393
2	Active	XENIX	400	1219	819

Total disk size: 1229 tracks (9 tracks reserved for masterboot and diagnostics).

There are two ways to switch operating systems once you have set up separate XENIX and DOS partitions:

- Use **fdisk** to change the current active partition.

To use **fdisk**, enter:

dos

at the boot prompt:

```
Boot
:
```

- Use a floppy diskette with the files necessary to boot the DOS operating system

If you change operating systems frequently, you should use a bootable DOS diskette to switch between DOS and XENIX. Follow this procedure:

1. Make sure all users are logged off XENIX.
2. Run **shutdown(ADM)** to shut down the XENIX system. This command makes sure all users know the system is being shut down, terminates all processes, then halts the system.
3. Once XENIX has shut down, insert the bootable DOS diskette into the primary (boot) drive.



4. Boot DOS.
5. To get back to XENIX, remove any disks from the floppy drive(s) and press **<CTRL><ALT>** (or turn the computer off, then on). Since the XENIX partition is still active, the XENIX operating system boots.

We recommend that you use a boot floppy or enter **dos** at your boot prompt to boot the DOS operating system. Booting from a floppy or the boot prompt is generally easier, faster, and safer than constantly using **fdisk** to change active partitions.

The other way to change operating systems is to run **fdisk** and change the active partition from XENIX to DOS. Then, after you shut down XENIX (see the previous steps) DOS boots from the hard disk. You do not need a bootable DOS floppy disk.

To switch back to XENIX, run **fdisk** under DOS and make the XENIX partition active. Then press (or turn the computer off, then **<CTRL><ALT>** on) to reboot XENIX.

Because the XENIX partition must be active for XENIX to operate, you cannot use a bootable floppy to boot XENIX. This second method is appropriate for an occasional change of the active operating system.

The following hard disk device names:

```
/dev/hd0d  
/dev/rhd0d  
/dev/hd1d  
/dev/rhd1d
```

are similar to */dev/hd0a* (the active disk partition) in that the disk driver determines which partition is the DOS partition and uses that as *hd?d*. This means that software using the DOS partition does not need to know which partition is DOS (the disk driver determines that).

Remember that if you have an active XENIX partition and boot DOS from a floppy you can transfer to C: to work with the DOS files.



9.3 Installing XENIX on a DOS System

If you wish to set up XENIX on a hard disk which previously contained only DOS, follow these steps:

1. Copy (back up) all the DOS files and directories on the hard disk onto floppies, or whatever backup media you wish to use.
2. Run **fdisk**, under DOS. If there is enough free space (at least 15 megabytes) for XENIX on your hard disk, skip to step 4. Otherwise, delete the DOS partition, then recreate it, leaving enough room on the disk for XENIX. Allow at least 6 megabytes for XENIX.
3. Return the DOS files to the hard disk from the backup media. Keep the backups in case there is an error of some kind, so you will not lose any data.
4. Turn off your computer.
5. Follow the installation procedure outlined in the *XENIX Installation Guide* to install XENIX.

You will see a message warning that the contents of the hard disk will be destroyed. Don't worry, you've backed up the DOS files and transferred them to the new DOS partition. The new partition being created will contain XENIX.

6. During the installation procedure **fdisk** is invoked to partition the hard disk. Use **fdisk** to assign a partition which is at least 15 megabytes to XENIX.
7. Designate "XENIX" as the active operating system.
8. Finish installing the XENIX operating system.

Note

XENIX **fdisk** displays DOS partitions as *DOS* while DOS **fdisk** displays XENIX partitions as *Other*.

You can only create DOS partitions using DOS **fdisk**, and only XENIX partitions using XENIX **fdisk**.

Be aware that DOS **fdisk** reports sizes in terms of cylinders, while XENIX **fdisk** reports sizes in terms of tracks. Check your hard disk manual for the number and size of cylinders on your hard disk.



9.4 Using XENIX and DOS With Two Hard Disks

Your computer always boots the operating system in the active partition on the first hard disk. XENIX must boot from the first hard disk. There are several ways to configure your system if you have two hard disks. Two ways are discussed here.

One configuration consists of designating the entire first disk as a XENIX partition. You then use a DOS boot floppy to start DOS and specify:

```
A> A: C:
```

to switch to the DOS area on the second hard disk, where **C** is the designation for the second hard disk. This strategy works for some versions of DOS. Early versions recognize only the first hard disk on the system.

Note

If you devote a hard disk for use with DOS, the disk must already be configured. See the "Adding A Second Hard Disk" section in the "Using Filesystems" chapter of this guide for details regarding hard disk configuration.

Another method is to maintain a small DOS partition on the first hard disk. The DOS partition is designated the active partition. In this configuration, the computer always boots DOS. This requires changing the active partition to boot the XENIX operating system from the hard disk.

If you use the entire second disk for DOS, you need only run **mkdev hd** to create device files for the second disk if you plan to use the XENIX DOS utilities (**doscp**, **dosls**, **doscat**, etc). If you do not wish to use those utilities to access DOS files on the second hard disk, there is no need to run **mkdev hd**.



Note

Be sure to make a backup copy of your boot floppies if you use them to boot your secondary operating system.

9.5 Removing an Operating System From the Hard Disk

You may find that you no longer need one of the operating systems installed on your hard disk. If you want to delete an operating system, use **fdisk** to delete the partition in question. Deleting the partition removes the contents of that partition and leaves unallocated space.

You can then reallocate that space by either adding another XENIX or DOS partition, or enlarging an existing partition. Enlarging a partition requires reinstalling the operating system and (for a XENIX partition) remaking the filesystem on the partition using **mkfs(ADM)**. Refer to the section on “Adding a Second Hard Disk” in the “Using Filesystems” chapter of this guide if you add a second XENIX partition and want to designate this partition as a mounted filesystem.

9.6 DOS Accessing Utilities

XENIX provides a set of utilities that help you bridge between the two operating systems. These are the XENIX commands, such as **dosls** and **doscat**, described in the XENIX manual page **dos(C)**. These programs allow you to access DOS files and directories which reside in a non-active DOS partition while running XENIX.

Note that you must have a bootable, although not active DOS partition on the hard disk or a DOS floppy in order to use these XENIX commands.

For example, you can only transfer a file from a XENIX partition on hard disk to a DOS floppy if either the DOS floppy is bootable or there is also a DOS partition on the hard disk.

Using these commands, you can list, copy, move and view the contents of DOS files and DOS directories. You may also be able to use the XENIX **dd(C)** and **diskcp(C)** commands to copy and compare DOS floppies. The XENIX **dtype(C)** command tells you what type of floppies you have (various DOS and XENIX types).



XENIX System Administrator's Guide

Also, the file */etc/default/msdos* describes which DOS filesystems (e.g. A:, B:, C: ...) correspond to which XENIX devices.

Note

You cannot execute (run) DOS programs or applications under XENIX.

If you have the XENIX Development System, with the *cmerge* compiler, you can create and compile programs that can be run under DOS operating systems. Refer to the *XENIX C User's Guide* appendix entitled "XENIX to MS-DOS: A Cross Development System" and the *C Library Guide* Appendix entitled "A Common Library for XENIX and MS-DOS" for more on using XENIX to create DOS programs. Also, see the DOS section in the *Programmer's Reference*.

9.7 XENIX and DOS On Non-Standard Disks

XENIX provides support for "non-standard" hard disks. The term "non-standard" refers to hard disks for which there are no correct disk parameter entries in your computer's ROM.

The correct parameters you specify for your non-standard disk(s) are stored in the masterboot block, which is the first sector of your boot hard disk drive. You can specify the hard disk characteristics during XENIX installation and these characteristics are then written out with the rest of the masterboot block. The special masterboot block resets the disk parameters to the specified values no matter which operating system is "Active". This mechanism provides non-standard disk support for both XENIX and DOS.

Although the special masterboot supports non-standard disks under DOS, you cannot use XENIX to install DOS on your hard disk. If a non-standard disk is being used, it is assumed that you already have some method to transfer your DOS files to the hard disk.



You should only use the XENIX **fdisk** to manipulate your hard disk partition table. Using DOS **fdisk** or custom **fdisks** provided by hard disk manufacturers after XENIX has been installed may disable non-standard disk characteristics, rendering your disk unusable.

Chapter 10

Preparing XENIX for Users

- 10.1 Introduction 10-1
- 10.2 Adding a User Account 10-1
- 10.3 Creating a Group 10-6
- 10.4 Changing a User's Login Group 10-7
- 10.5 Changing a User ID 10-9
- 10.6 Removing a User Account 10-11
- 10.7 Changing XENIX Initialization 10-13
 - 10.7.1 Changing the /etc/rc File 10-14
 - 10.7.2 Changing the .profile and .login Files 10-15
 - 10.7.3 Changing the /etc/motd File 10-15



10.1 Introduction

User accounts help the XENIX system administrator keep track of the people using the system and control their access to system resources. Ideally, each user should have a user account. Each account has a unique “login name” and “password” with which the user enters the system, and a “home directory” where the user works.

It is the system administrator’s job to create accounts for all users on the system, and maintain these accounts by changing user passwords, login groups, and user IDs when necessary.

This chapter explains how to:

- Add user accounts to the system
- Create a group
- Change an account’s login group
- Change an account’s user ID
- Remove user accounts from the system

The following sections describe each task in detail.

10.2 Adding a User Account

You can add a user account to the system with the **mkuser** program. The program creates a new entry in the XENIX system’s */etc/passwd* file. This entry contains information about the new user (such as login name and initial password) that the system uses to let the user log in and begin work. The program also creates a home directory for the user, a mailbox for use with the **mail** command, and an initialization file (for example, *.profile* for the Bourne shell or *.login* for C-shell) containing XENIX commands that are executed when the user logs in.

To create a new user account, follow these steps:

1. Log in as the super user.

2. Enter:

mkuser

Δ sysadmsh users select: Users→Add

and press the **RETURN** key. The system displays the following message:

```
Mkuser
Add a user to the system
Do you require detailed instructions? (y/n/q)
```

3. Enter the letter *y* (for “yes”), if you want information about the program, otherwise type the letter *n* (for “no”). Enter *q* (for “quit”) only if you wish to stop the program and return to the system. If you type a “q” to any “(y/n)” prompt, the program will stop and no changes will be made.
4. When the program continues, you are prompted for user id:

```
Do you wish to use the next available user id? (y/n/q)
```

If you enter *n*, you are asked to specify the id number you wish to use.

5. Next, you are prompted for the login name:

```
Enter new user's login name, or enter q to quit:
```

The login name is the name by which XENIX identifies the user. It is usually a short version of the user's actual name, typed in lower-case letters. For example, either “johnd” (a first name and last initial) or “jdoe” (a first initial and last name) is customary for the user John Doe.

6. Enter the new name, and press the **RETURN** key. The program now prompts you for information about the new user's group name and group number.

A group name is the name of the group of users to which the new user will belong. Users in a group have access to a common set of files and directories. The group name is optional.

The program prompts:

```
Do you want to use the default group? (y/n):
```

If you enter "y", the user's group name will be "group" and the group ID number will be 50.

If you enter "n", the program responds with a list of existing groups:

```
Existing groups are:
```

```
Group 'group' (50): demo vdemo cdemo
```

```
Do you want to use one of these groups? (y/n):
```

If you enter either "y" or "n", you are asked which group you want to use. Enter the name of the group. You may create a new group by entering in the new name.

Next, you are prompted for a group number. The group ID, or number, may be any number from 50 to 30000 that isn't already used for another group.

7. After entering the group name and ID, you are prompted for the initial password.

```
Enter at least 5 characters for the password.  
Enter password:
```

The initial password is the password you assign to the new user.

The user will use the initial password to enter the account for the first time. Once in the account, the user should create a new password for himself, one that is hard to guess. (See the section "Changing Your Password" in the "Logging In" chapter of the *XENIX Tutorial*.)

8. Enter the password, and press the **RETURN** key.
9. Next, you are prompted for a shell type. You see a list and brief explanation of the available shells (the menu selections depend on what packages and/or applications installed on your system) and the prompt:

```
ENTER Shell type (1, 2, 3,...) and press RETURN:
```

sh is the standard (Bourne) shell. **vsh** is the menu driven "visual" shell, **csh** is the C-shell, **rsh** is the restricted shell, and **uucp login** is an entry in */usr/lib/uucp/uucico* enabling logging in to the system via **uucp**. For more information, see **sh(C)** and **csh(C)** in the *XENIX User's Reference*.

10. Enter the desired shell number and press **RETURN**. After you have entered the shell type, the program prompts you for a comment:

```
Please enter Comment  >-----  
>
```

A comment is information about the new user, such as a department name and phone extension. Although, the comment is optional, it is useful if the **finger(C)** command is often used to display information about users. If given, the comment must be no more than 20 characters long, including spaces. It must not contain any colons (:). The example

```
John Doe, 123
```

shows the recommended form for a comment.

11. Enter the comment. Make sure it is 20 characters or less. If you do not wish to enter a comment, just press **RETURN**.

The program now displays what you have entered and the special user entry that it has created for the new user. This entry is copied to the special system file */etc/passwd*. The entry shows the login name, the encrypted password, the user ID, the group ID, the comment, the user's home directory, and the startup program. Fields in the entry are separated by colons (:). (For a full description of each field, see **passwd(F)** in the *XENIX User's Reference*.)

The program then gives you an opportunity to change the user name, password, group, or comment:

```
Username is "johnd", user ID is 2001.  
Group name is "group", group number is 50.  
Comment field is: John Doe, 123  
Shell is "/bin/csh"
```

Do you want to change anything? (y/n):

12. Enter the letter **y** (for "yes") and press the **RETURN** key, if you wish to change something. Enter **n** (for "no") and skip to the next step if you wish to complete the new account. (Enter **q**, for "quit", only if you wish to leave the program and abort the new account.)

If you enter **y**, the program prompts for the field you wish to change:

```
1. User name  
2. User id (uid)  
3. Group  
4. Shell  
5. Password  
6. Comment in /etc/passwd
```

Select an item to change, or enter **q** to quit:

Enter the name of the item (field) you wish to change and press **RETURN**. After you have changed a field, you see the complete list of fields and are asked if you wish to make other changes. When you are finished with any changes, the program adds the user.

13. The program displays the messages:

```
Password file updated.  
Group file updated.  
Home directory /x/name created.  
/x/name/.shellfile created.  
Test mail sent to user user.  
User name added to this system.
```

The program then asks if you wish to add another user to the system.

14. Enter **y** if you wish to add another user. Otherwise, enter **n** to stop the program and return to the super user prompt.

A user can log into a new account as soon as it is created. For details see the "Logging In" chapter of the *XENIX Tutorial*.

10.3 Creating a Group

A group is a collection of users who share a common set of files and directories. The advantage of groups is that users who have a common interest in certain files and directories can share these files and directories without revealing them to others. Initially, all users belong to the common system group named "group", but you can create new groups by modifying the XENIX system file */etc/group* using a XENIX text editor.

To create a new group, you need to choose a group name and a group identification number (group ID). You also need to make a list of the users in the new group. The group name may be any sequence of letters and numbers up to eight characters long, and the group ID may be any number in the range 50 to 30000. Both the group name and ID must be unique, i.e., they must not be the same as any existing group name or ID.

To create a new group, follow these steps:

1. Log in as the super user.
2. Display the contents of the */etc/group* file by entering:

```
cat /etc/group
```

and pressing the **RETURN** key. The **cat** command displays the contents of the */etc/group* file. The file contains several entries, each defining the group name, group ID, and users for a group.

Each entry has the form:

```
group-name::group-ID:users
```

The users are shown as a list of login names separated by commas (.). For example, a typical file may look like this:

```
other:x:1:demo
sys:x:2:
group::50:johnd,suex
```

3. Check the */etc/group* file entries to see that the group name and ID you have chosen are unique.
4. If the group name and ID are unique, invoke a XENIX text editor (see the *XENIX User's Guide*) and specify */etc/group* as the file to edit.
5. Locate the last line in the file, then insert the new entry in the form given above. For example, if you wish to create a group named "shipping" with group ID "142" and users "johnd", "marym", and "suex", enter:

```
shipping::142:johnd,marym,suex
```

6. Exit the editor.

To make sure you have entered the group names correctly, use the **grpcheck(C)** command to check each entry in the */etc/group* file. If the new entry is free of errors, no other changes to the file are required.

You can create any number of new groups. Each group may have any number of members. Furthermore, any user may be a member of any number of groups. Multiple group membership is especially convenient for users who have interests that span a variety of areas.

If a user is a member of several groups, they can gain access to each group by using the **newgrp(C)** command.

10.4 Changing a User's Login Group

When a user logs in, the system automatically places the user in the proper "login group". This is the group given by the group ID in the user's */etc/passwd* file entry (see the section "Adding a User Account" in this chapter). You can change the user's login group by changing the group ID. To change the group ID you need the group ID of the new login

XENIX System Administrator's Guide

group, and you need to know how to use a XENIX text editor (see the *XENIX User's Guide*).

To change the group ID, follow these steps:

1. Log in as the super user.

△ **sysadmsh** users select: Users→Modify→Group

2. Use the **cd** command to change the current directory to the */etc* directory. Enter:

```
cd /etc
```

3. Use the **cp** command to make a copy of the */etc/passwd* file. Enter:

```
cp passwd passwd+
```

4. Invoke a text editor and specify */etc/passwd+* as the file to edit.
5. Locate the desired user's password entry. Each entry begins with the user's login name.
6. Locate the user's group ID number in the user's password entry. It is the fourth field in the entry. Fields are separated by colons (:). For example, the following entry has group ID "50":

```
marym:9iK1wp:205:50:Mary March,122:/usr/marym:/bin/sh
```

7. Delete the old group ID and insert the new one. Be sure you do not delete any other portion of the user's password entry.
8. Exit the editor.
9. Use the **mv** command to save the old */etc/passwd* file. Enter:

```
mv passwd passwd-
```

10. Use the **mv** command to make the edited file the new */etc/password* file. Enter:

```
mv passwd+ passwd
```

You can make sure you have entered the new login group correctly by using the **pwcheck(C)** command. If the new entry is correct, no other changes to the file are required.

You must not change the group IDs for system accounts such as “cron” and “root”. System accounts are any accounts with user IDs less than 200. The user ID is the third field in the password entry.

Note that changing a user’s login group does not change the “group ownership” of the files. Group ownership defines which group has access to a user’s files. If users in the new group wish to access the user’s files, you must change the group ownership with the **chgrp(C)** (for “change group”) command. For details, see the section “Changing Group Ownership” in the “Using Filesystems” chapter.

10.5 Changing a User ID

Sometimes it is necessary to change the user ID in a user’s account entry to allow a user to access files and directories transferred from other computers. In particular, if a user has different accounts on different computers and frequently transfers files and directories from one computer to another, the user IDs in each of the account entries must be made the same. You can make them the same by modifying the account entries in the */etc/passwd* file.

To change a user ID, follow these steps at every computer for which the user has an account:

1. Log in as the super user.
2. Use the **cd** command to change the current directory to the */etc* directory. Enter:

```
cd /etc
```

3. Use the **cp** command to make a copy of the */etc/passwd* file. Enter:

```
cp passwd passwd+
```

4. Invoke a XENIX text editor and specify */etc/passwd+* as the file to edit.
5. Locate the user’s account entry. Each entry begins with the user’s login name.

XENIX System Administrator's Guide

6. Locate the current user ID. The ID is the third field in the entry. For example, the following entry has a user ID of "205":

```
marym:9iK1wp:205:50:Mary March,122:/usr/marym:/bin/sh
```

Substitute the new user ID for the old one.

7. Exit the text editor.
8. Use the **mv** command to save the old */etc/passwd* file. Enter:

```
mv passwd passwd-
```

9. Use the **mv** command to make the edited file the new */etc/passwd* file. Enter:

```
mv passwd+ passwd
```

No other changes to the file are required.

In most cases, you can change the user ID to the same number as the user's most-used account. But the new number must be unique on every system for which the user has an account. If there is any conflict (for example, if the number already belongs to another user on one of the systems), you must choose a new number. You can choose any number greater than 200. Make certain it is unique, and that you copy it to all systems on which the user has an account.

Once a user's ID has been changed, you must change the "user ownership" of the user's files and directories from the old user ID to the new one. You can do this with the **chown(C)** (for "change owner") command described in the "Maintaining System Security" chapter.

For example, to change the ownership of johnd's home directory, enter:

```
chown johnd /usr/johnd
```

Note that you may use the **find(C)** command described in "Finding Files" in the "Working with Files and Directories" chapter of the *XENIX Tutorial* to locate all files and directories with the user's old user ID.

10.6 Removing a User Account

It is sometimes necessary to remove a user account from the system. You can remove a user account with the **rmuser**(ADM) program. The program deletes the user's entry from the */etc/passwd* file and removes the user's home directory and mailbox.

Before you can remove the user account, you must remove all files and directories from the user's home directory, or move them to other directories. If you wish to save the files, you can use the **tar**(C) command to copy the files to a floppy disk (see "Making Backups" in the "Housekeeping" chapter of the *XENIX Tutorial*.)

To remove a user account, follow these steps:

1. Log in as the super user.
2. Enter:

```
cd /usr/login-name
```

and press the **RETURN** key to change to the user's home directory. The *login-name* must be the user's login name.

3. Make sure that you have made copies of all important files and directories in the user's home directory.
4. Use the **rm** (for "remove") command to remove all files and directories from the user's home directory. This includes any files that begin with a period (.). Directories can be removed by using the **-r** (for "recursive") option of the **rm** command. For example, the command:

```
rm -r bin
```

removes the directory named *bin* and all files within this directory.

5. After removing all files and directories, make sure the user's mailbox is empty. Enter the following command, substituting the user's login name for *login-name*:

```
cat /usr/spool/mail/login-name
```

If the mailbox contains text, enter:

```
cat /dev/null > /usr/spool/mail/login-name
```

and press the **RETURN** key.

6. When the user's home directory and mailbox are empty, enter:

```
cd /usr
```

and press the **RETURN** key. The user's home directory cannot be removed until you have moved to another directory.

7. Enter the following command, followed by **RETURN**:

```
rmuser
```

△ **sysadmsh** users select: Users→Delete

The program displays a message explaining how to remove a user:

```
rmuser
remove a user from the system
Press RETURN when you are ready.
```

The program then prompts you for the login name of the user you wish to remove:

```
Enter name of id to be removed.
```

8. Enter the user's login name. You should now see the message:

```
Removing user name from the system. CONFIRM? (y/n/q):
```

9. Enter **y** (for “yes”) to remove the user from the system. Otherwise enter **n** (for “no”) to stop the removal, or **q** (for “quit”) to stop the program. The program removes the user’s entry from the */etc/passwd* file, the user’s mailbox, *.profile* file, and home directory. The program displays the message:

```
User name removed from the system
```

The program now gives you a chance to remove another user:

```
Do you want to remove another user? (y/n/q):
```

10. Enter **y** to remove another user. Otherwise, enter **n** or **q** to stop the program.

Note that the **rmuser** program will refuse to remove an account that has a system name, such as “root”, “sys”, “sysinfo”, “cron”, “uucp”, or a system ID (user ID below 200). Also, the program cannot remove a user account if the user’s mailbox still has mail in it, or if the user’s home directory contains files other than *.profile*.

10.7 Changing XENIX Initialization

When your system is switched on and booted, the certain aspects of system operation are initialized, including the mounting of filesystems. You can adapt system initialization by modifying the system initialization files.

The XENIX initialization files contain XENIX commands and/or data which the system reads at system startup or whenever a user logs in. The files typically mount filesystems, start programs, and set home directories and terminal types. The initialization files are named */etc/rc*, *.profile*, and */etc/motd*.

The system administrator can modify these files to create any desired initial environment. The files are ordinary text files and may be modified

using a text editor such as **vi(C)** (see the *XENIX User's Guide*). Note, however, that the */etc/rc* and *.profile* files contain XENIX commands and comments, and have the command file format described in "The Shell," chapter in the *XENIX User's Guide*.

10.7.1 Changing the */etc/rc* File

The */etc/rc* file contains XENIX system initialization commands. The system executes the commands at system startup. The commands display a startup message, start various system daemons, and mount filesystems. You can display the contents of the file with the **more(C)** command. Enter:

```
more /etc/rc
```

and press the **RETURN** key.

You may change the contents of the file so that the system executes any set of commands you wish. For example, if you want the system to automatically mount a new filesystem, simply append the appropriate **mount** command in the file. The system will execute the command on each startup.

To append a command to the file, follow these steps:

1. Log in as the super user.
2. Invoke a text editor and specify the */etc/rc* as the file to be edited.
3. Locate the place in the file you wish to insert the command (e.g., if the command mounts a filesystem, insert it with other mounting commands).
4. Insert the command on a new line. Make sure you enter the command correctly. The system rejects any incorrect commands and the commands that follow it when the file is read at system startup.
5. Exit the editor.

No other changes to the file are required. Be careful not to delete any commands already in the file unless you are sure they are not needed.

10.7.2 Changing the *.profile* and *.login* Files

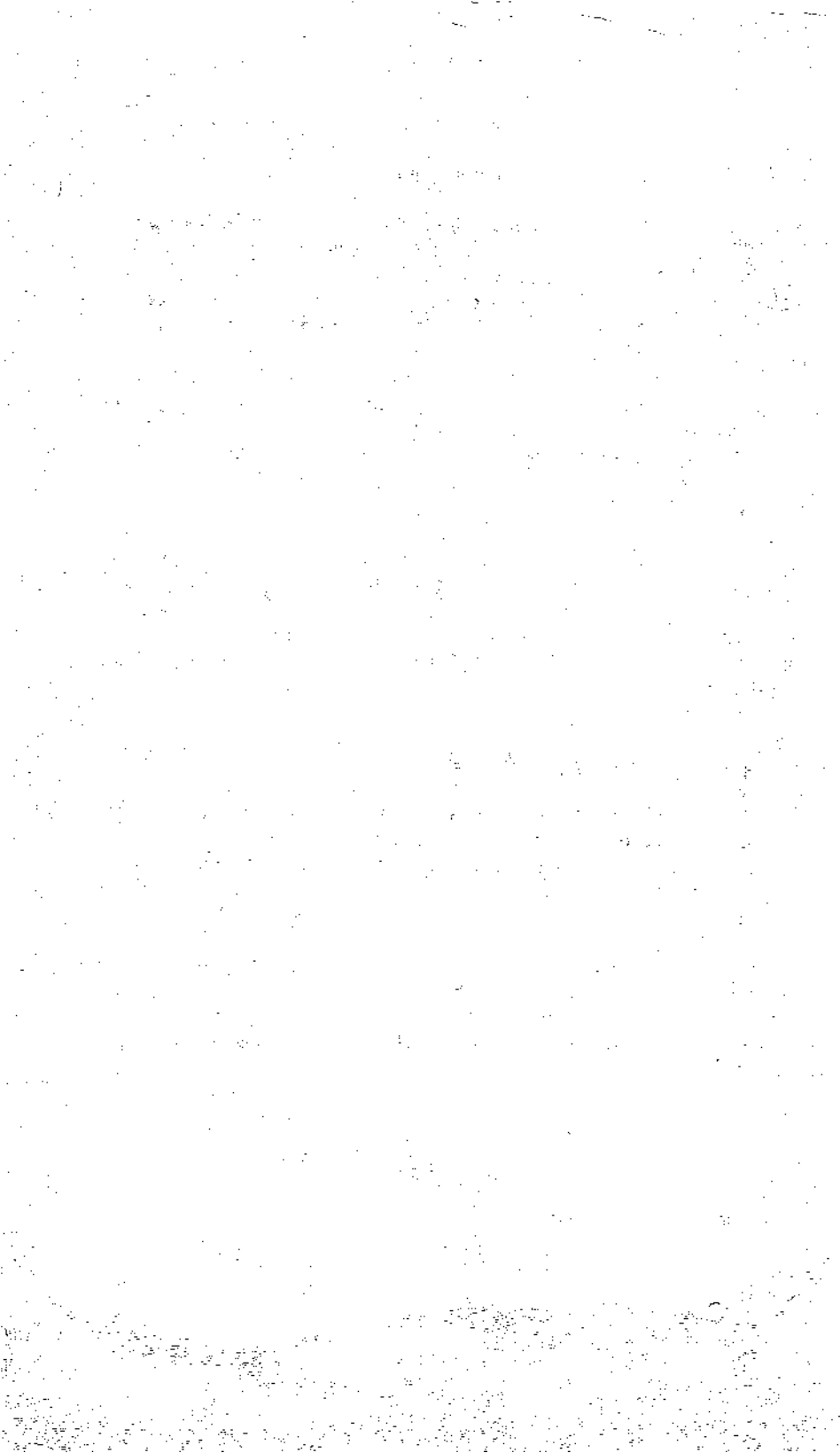
The *.profile* and *.login* files contain commands that initialize the environment for each user. If the user uses the standard command shell */bin/sh*, they have the file *.profile* in their home directory. If the user uses the C-shell */bin/csh*, they will have the file *.login* in their home directories. (Depending on the login shell, other files may apply.) The commands in these files are executed whenever the user logs in. The files usually contain commands that set various system variables (e.g., *TERM*, *PATH*, *MAIL*). These variables give the system information such as what terminal type is being used, where to look for programs the user runs, where to look for the user's mailbox, what keys to expect for the "kill" and "backspace" functions, and so on. (see the chapters about shell and C-shell in the *XENIX User's Guide*).

There is one *.profile* or *.login* file for each user account on the system. The files are placed in the user's home directory when the account is created. An ordinary user may modify their own *.profile* or *.login* file or allow the system manager to make modifications. In either case, the file can be edited like the */etc/rc* file by using a text editor. Commands can be added or removed as desired.

10.7.3 Changing the */etc/motd* File

The message of the day file, */etc/motd*, contains the greeting displayed whenever a user logs in. Initially, this file contains the name and version number of the XENIX system. It can be modified to include messages such as a reminder to clean up directories, a notice of the next periodic backup, and so on.

The */etc/motd* file is an ordinary text file, so you can change the message by editing the file with a text editor. One common change is to include a reminder to delete unused files in order to preserve disk space. In general, you should limit the size of the file to include no more than a screenful of information.



Chapter 11

Building a Remote Network with UUCP

- 11.1 Introduction 11-1
 - 11.1.1 What is UUCP? 11-1
 - 11.1.2 How to Use This Guide 11-2
 - 11.1.3 What You Need 11-2
 - 11.1.4 UUCP Commands 11-3
 - 11.1.5 UUCP Directories 11-5
 - 11.1.6 UUCP Background Programs 11-6
 - 11.1.7 How UUCP Works 11-6
 - 11.1.8 A Sample UUCP Transaction 11-7
- 11.2 Connecting Two Local Systems Using A Direct Wire 11-9
 - 11.2.1 Choose a Serial Line 11-9
 - 11.2.2 Connect a Serial Wire 11-10
- 11.3 Connecting Remote UUCP Systems with a Modem 11-11
 - 11.3.1 Choose a Serial Line 11-12
 - 11.3.2 Set the Dialing Configuration 11-13
 - 11.3.3 Connect the Modem 11-13
 - 11.3.4 Configuring a Hayes 2400 or Compatible Modem 11-14
 - 11.3.5 Variable Rate Modems 11-16
 - 11.3.6 Test the Modem 11-16
- 11.4 Configuring UUCP on Your System 11-17
 - 11.4.1 An Important Consideration: Call or Be Called? 11-18
 - 11.4.2 Setting Up the Control Files with `uinstall` 11-18
 - 11.4.3 Establish Your Site Name in the `/etc/systemid` File 11-20
 - 11.4.4 Selecting and Defining a UUCP port 11-21
 - 11.4.5 Creating Login Accounts for Sites Dialing-in 11-23
 - 11.4.6 Adding Entries for Remote Sites to the Systems File 11-24
 - 11.4.7 Limiting Access with the Permissions File 11-30
 - 11.4.8 Adding Dial-out Entries to the the Devices File 11-38
 - 11.4.9 Using the Same Port for Dialing In and Out 11-44

- 11.5 Special UUCP Configuration Options 11-44
 - 11.5.1 Adding Dialers to the Dialers File 11-44
 - 11.5.2 Using Dialcodes to Create a Portable Systems File 11-46
 - 11.5.3 Using a Streams-based Network: the DevconfigFile 11-48
 - 11.5.4 Creating Alternate Control Files with Sysfiles 11-48
 - 11.5.5 Preventing Unknown Sites from Logging in 11-49
 - 11.5.6 Linking Micnet Sites with the UUCP Network 11-50

- 11.6 Administrating Your UUCP System 11-50
 - 11.6.1 Scheduling Communications with Other Systems 11-50
 - 11.6.2 Automated Maintenance 11-52
 - 11.6.3 Generating Log Reports on UUCP Usage: uulog 11-54
 - 11.6.4 The UUCP Spool Directory 11-55

- 11.7 Troubleshooting 11-57
 - 11.7.1 Check for Faulty ACU/Modem 11-58
 - 11.7.2 Check the Systems File 11-58
 - 11.7.3 Debug Transmissions 11-58
 - 11.7.4 Check Basic Information 11-59

- 11.8 Keeping Traffic and Congestion Under Control 11-59
 - 11.8.1 Crowded Directories and Lack of Space 11-60
 - 11.8.2 Running out of Processes 11-60
 - 11.8.3 Evaluating Apparent Stoppages 11-60

- 11.9 UUCP Error Messages 11-61
 - 11.9.1 ASSERT Error Messages 11-61
 - 11.9.2 UUCP STATUS Error Messages 11-63

11.1 Introduction

This chapter explains how to use the UUCP package to build a remote network system for your computer using a normal telephone line and a modem.

UUCP can also be used to physically connect dissimilar XENIX/UNIX systems (for example, two systems that lack a common local area network program, like **micnet**). using a direct serial line. In addition, the UUCP system is a practical solution to the problem of two **micnet** networks that cannot be connected because of distance or cost of cable (See the chapter on “Building a Local Network with Micnet” in this guide.)

11.1.1 What is UUCP?


The UUCP package permits XENIX/UNIX systems to communicate as part of a remote network. The name UUCP is an acronym for “UNIX-to-UNIX Copy.” The UUCP package consists of a group of programs that provide the following capabilities:

- Remote file transfer (**uucp**)
- Remote command execution (**uux**)
- Delivery/forwarding of mail to remote sites (via **mail**)

Used primarily over phone lines, UUCP can be used to connect with specific remote machines on a demand or scheduled basis and by either dialing out, or allowing other machines to dial in.

UUCP uses a batch method to manage communication traffic, storing (or “spooling”) requests for later execution when actual contact is made between systems. Thus, when UUCP commands are executed by the user, work files and any data files needed are created in */usr/spool/uucp*. The program *uucico* scans this directory for the instructions contained in any work files and executes them. Although it is possible to execute commands immediately, most systems are set up to connect with other systems according to a daily schedule (usually during the evenings to reduce connection costs).

11.1.2 How to Use This Guide



This guide describes how to build a UUCP system and covers both hardware installation and software configuration. There are also sections on routine maintenance and troubleshooting.

The following is an outline of what must be done to set up your UUCP network:

1. Connect and configure a modem or direct wire.
2. Configure the UUCP software using **uinstall**.
3. Create login accounts for any sites that will be calling your system.
4. Test your connections with each remote site.

The most important task of configuring UUCP is the editing of several control files that act as the database for UUCP. The next few sections describe the function of these files, and "Configuring UUCP on Your System" explains the information that these files contain. The **uinstall** utility will edit these files for you and explain each entry. **uinstall** also includes an extensive help facility. Read "Configuring UUCP on Your System" carefully before running **uinstall** to understand the UUCP database.

11.1.3 What You Need

To install a direct wire communication system on your computer, you need

- At least one RS-232 serial line (or serial port) on your computer to use for UUCP.
- The UUCP programs and files extracted from your XENIX System distribution.

If you want to use your computer as a dial-in and/or dial-out site with a modem, you also need:

- A modem. Supported modems include models by Hayes, Penril, Ventel, Vadic, Rixon, AT&T, and Telebit; but you can supply *Dialers* entries or dialer programs for other modems. Instructions for the Hayes Smartmodem 1200 or 2400 and compatibles are given below.

- A standard telephone jack for access to the telephone system (touch tone line required for the Hayes).
- An RS-232 cable to connect the serial line to the modem.



11.1.4 UUCP Commands

UUCP programs are divided into two categories: user programs and administrative programs. The paragraphs that follow describe the programs in each category.

User Programs

The user programs for basic networking are in `/usr/bin`. No special permission is needed to use these programs. These commands are all described in the “Communicating with Other Sites” chapter of the *XENIX User’s Guide*.

- | | |
|-------------|---|
| cu | Connects your computer to a remote computer so you can be logged in on both at the same time, allowing you to transfer files or execute commands on either computer without dropping the initial link. |
| ct | Connects your computer to a remote terminal so the user of the remote terminal can log in. The user of a remote terminal can call the computer and request that the computer call it back. In this case, the computer drops the initial link so that the remote terminal’s modem will be available when it is called back. |
| uucp | Lets a user copy a file from one computer to another. It creates work files and data files, queues the job for transfer, and calls the uucico daemon, which in turn attempts to contact the remote computer. |
| uuto | Copies files from one computer to a public spool directory on another computer (<code>/usr/spool/uucppublic/receive</code>). Unlike uucp , which lets you copy a file to any accessible directory on the remote computer, uuto places the file in an appropriate spool directory and tells the remote user to pick it up with uupick . |



- uupick** Retrieves the files placed under */usr/spool/uucppublic/receive* when files are transferred to a computer using **uuto**.
- uux** Creates the work, data, and execute files needed to execute commands on a remote computer. The work file contains the same information as work files created by **uucp** and **uuto**. The execute files contain the command string to be executed on the remote computer and a list of the data files. The data files are those files required for the command execution.
- uustat** Displays the status of requested transfers (**uucp**, **uuto**, or **uux**). It also provides you with a means of controlling queued transfers.

Administrative Programs

Most of the administrative programs are in */usr/lib/uucp*, along with the control files and shell scripts. The only exception is **uulog**, which is in */usr/bin*. These commands are described in the (ADM) section of this Guide.

- uulog** Displays the contents of a specified computer's log files. Log files are created for each remote computer your computer communicates with. The log files contain records of each use of **uucp**, **uuto**, and **uux**.
- uucleanup** Cleans up the spool directory. It is normally executed from a shell script called **uudemon.clean**, which is started by **cron**.
- uutry** Tests call processing capabilities and does a moderate amount of debugging. It invokes the **uucico** daemon to establish a communication link between your computer and the remote computer you specify.
- uucheck** Checks for the presence of basic networking directories, programs, and support files. It can also check certain parts of the *Permissions* file for obvious syntactic errors.

uinstall Allows the UUCP control files to be configured with a simple menu interface. Also configures ports and can be used to convert Version 2 UUCP control files to Honey DanBer format.

11.1.5 UUCP Directories

There are three directories associated with UUCP:

- /usr/spool/uucp* This is the working directory for UUCP. Work files, log files and all UUCP communication traffic are stored here.
- /usr/spool/uucppublic* This is the publically-readable/writable target directory used for most file transfers.
- /usr/lib/uucp* Most of the UUCP programs are stored here, as well as the supporting database/control files. The main user programs, including **uux** and **uucp**, are found in */usr/bin*.

In addition to programs, */usr/lib/uucp* contains configurable data files for use by UUCP (easily distinguished by their capitalized names). Of these, the most important to understand are:

- Systems* Contains information needed to establish a link to a remote computer. It contains information such as the name of the remote computer, the name of the connecting device associated with the remote computer, when the computer can be reached, telephone number, login ID, and password.
- Permissions* This file defines the level of access that is granted to computers when they attempt to transfer files or remotely execute commands on your computer.
- Devices* Contains information concerning the location, line speed, and type of the automatic call units, direct links, and network devices.

11.1.6 UUCP Background Programs



uucp traffic is managed by three *daemons*, or supervisory programs that run in the background, handling file transfers and command executions. (The daemons can also be executed manually as commands.)

- uucico** Selects the device used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers data and execute files, logs results, and notifies the user by **mail** of transfer completions. When the local **uucico** daemon calls a remote computer, it “talks” to the **uucico** daemon on the remote computer during the session.

- uuxqt** Executes remote execution requests. It searches the spool directory for execute files (always named *X.file*) that have been sent from a remote computer. When an *X.file* file is found, **uuxqt** opens it to get the list of data files that are required for the execution. It then checks to see if the required data files are available and accessible. If the files are present and can be accessed, **uuxqt** also verifies that it has permission to execute the requested command.

- uusched** Schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** randomizes the order in which remote computers will be called.

11.1.7 How UUCP Works

When you enter a UUCP command, the program creates a work file and usually a data file for the requested transfer. The work file contains information required for transferring the file(s). The data file is simply a copy of the specified source file. After these files are created in the spool directory, the **uucico** daemon is started.

The **uucico** daemon attempts to establish a connection to the remote computer that is to receive the file(s). It first gathers the information required for establishing a link to the remote computer from the *Systems* file. This is how **uucico** knows what type of device to use in establishing the link.

Then **uucico** searches the *Devices* file looking for the devices that match the requirements listed in the *Systems* file. After **uucico** finds an available device, it attempts to establish the link and log in on the remote computer.



When **uucico** logs in on the remote computer, it starts the **uucico** daemon on the remote computer. The two **uucico** daemons then negotiate the line protocol to be used in the file transfer(s). The local **uucico** daemon then transfers the file(s) that you are sending to the remote computer. The remote **uucico** places the file in the specified path name(s) on the remote computer. After your local computer completes the transfer(s), the remote computer may send files that are queued for your local computer. The remote computer can be denied permission to transfer these files with an entry in the *Permissions* file. If this is done, the remote computer must establish a link to your local computer to perform the transfers.

If the remote computer or the device selected to make the connection to the remote computer is unavailable, the request remains queued in the spool directory. Each hour (default), **uudemon.hour** is started by **cron** which in turn starts the **uusched** daemon. When the **uusched** daemon starts, it searches the spool directory for the remaining work files, generates the random order in which these requests are to be processed, and then starts the transfer process (**uucico**) described in the previous paragraphs.

11.1.8 A Sample UUCP Transaction

The following traces the execution of a **uucp** command:

1. A user on a system called “kilgore” wishes to send a copy of the file “minutes.01.10” to a remote system called “obie”. To accomplish this, the user enters the following command:

```
uucp minutes.01.10 obie\usr/spool/uucppublic
```

Note that the exclamation point need only be escaped (preceded by a “\”) if the **cs**h is used; the Bourne shell (**sh**) doesn’t require this.

2. A work file is created in the *usr/spool/uucp/obie* directory, *c.obienxxxx*.
3. The **uusched** daemon schedules the request for execution by **uucico**.
4. When the execution time is reached, **uucico** first checks the *Systems* file and confirms that *obie* is a recognized system and that a call is permitted at this time.



5. Using the information in the *Systems* file, **uucico** next locates the modem device and tty port associated with it as stored in the *Devices* file.
6. Using the phone number in the *Systems* file and the modem type from the *Devices* file, **uucico** uses the appropriate modem commands from the *Dialers* file (or alternately runs a dialer program from the *lib* directory) to connect to the remote system.

UUCP Control Files (sites: <i>kilgore</i> and <i>obie</i>)	
<i>Systems:</i>	obie Any ACU 2400 14081234567 \ ogin:--ogin: nuucp ssword: mavra
<i>Devices:</i>	ACU tty11 - 2400 hayes2400
<i>Permissions:</i>	LOGNAME= ukilgore MACHINE= kilgore \ READ=/usr/spool/uucppublic:/usr/kilgore \ WRITE=/usr/spool/uucppublic:/usr/kilgore \ REQUEST=n SENDFILES=call

7. **uucico** creates a lock file to lock the serial line in the directory */usr/spool/uucp*.
8. **uucico** uses the login sequence and password defined in the *Systems* file to login to "obie", whose own **uucico** confirms that "kilgore" is recognized before beginning the actual transaction.
9. The calling system, "kilgore," (sometimes known as the *guest*) is said to be the "master" of the transaction; the called system, "obie," (also known as the *host*) is said to be the "slave." The slave **uucico** checks the local *Permissions* file to confirm that the master is authorized to transfer the file.
10. The master ("kilgore") transmits the file in packets that are checked for errors and retransmitted if garbled. During reception, the file is stored in a temporary file in the */usr/spool/uucp* directory. When the transfer is complete, the file is transferred to the proper destination, in this case */usr/spool/uucppublic/minutes.01.10*.

11. Each machine records its side of the transaction in logfiles. (For example, “obie” would have the exchange recorded in a file called */usr/spool/uucp/Log/uucp/kilgore.*)
12. Unless the slave system (“obie”) has requests of its own, a hangup request is sent, the connection is terminated, and the lock removed.



In the case of remote command execution (via **uux**), an execute x.file is created in the */usr/spool/uucp* directory. The **uuxqt** daemon scans this directory for work, checks the *Permissions* file to confirm permission to execute the command, then executes it.

Having explained how UUCP functions, the next step is to set up the programs to work on your system.

11.2 Connecting Two Local Systems Using A Direct Wire

This section describes how to install a direct wire between two computers. If you are using UUCP to connect remote machines, you can skip this section. To connect two computers with a direct wire, you need to:

- Choose a serial line on each machine.
- Connect a serial wire (RS-232) between the two machines, using the chosen serial lines.
- Decide which machine is the dial-in site and which is the dial-out site. The dial-out site calls up and logs in to the dial-in site.

When you finish with these steps, you can proceed with the next sections to set up the sites.

11.2.1 Choose a Serial Line

On each machine, you must choose the RS-232 serial line you want to use. If there are no lines available, you must install a new serial line or make one available by removing any device connected to it. If you remove a terminal, make sure no one is logged in.

Once you have chosen a serial line, find the name of the device special file associated with the line by referring to the "XENIX Directories and Special Device Files" chapter in this guide. The filename should have the form

`/dev/tty nn`

where nn is the number of the corresponding line. For example, `/dev/tty1a` usually corresponds to COM1. You need the name of the actual line for later steps.

The serial line you use for your communication system should be owned by **uucp**. To make sure the line is owned by **uucp** enter this command:

chown uucp /dev/tty nn

where nn is the number of the corresponding line.

11.2.2 Connect a Serial Wire

You connect two computers together using an RS-232 cable. The actual pin configurations sometimes vary between machines.

Typically, the wire should connect pins 2, 3, and 7 (and/or 20) on one computer to the same pins on the second computer. Sometimes the cable must be *nulled*, which means that pin 2 on one machine is connected to pin 3 on the other, and vice versa.

Since the connections can vary, you should check the hardware manuals for each computer to determine the proper pin connections.

Testing a Connection

For this section, `tty2a` is used as the example serial line for both machines.

To test the wire connection between two machines, follow these steps:

1. Disable the serial lines on each machine. On each computer, enter the command:

disable /dev/tty2a

Be sure to disable the modem control line as well:

disable /dev/tty2A

2. Attach one end of the serial wire to one of the machines. Attach the other end to the standard data port of a terminal.
3. Enter this command at the computer:

```
(stty 9600; date ) < /dev/tty2a > /dev/tty2a
```

tty2a is our example serial line, and the **date** command provides sample output.

You should see the output of the **date** command appear on the terminal screen. Repeat this procedure on the other machine.

If this doesn't work, check the following:

- The wire is plugged in properly at each end.
- The continuity of the wire.
- The terminal is configured correctly (baud rate, parity, etc.).
- The serial line is disabled.
- You are using the correct pin numbers.

Note

An unterminated serial line can cause serious system problems. Do not leave serial lines dangling.

11.3 Connecting Remote UUCP Systems with a Modem

With a modem, you can communicate with computers over standard phone lines. These are the steps to install a modem:

- Choose a serial line.
- Set the dialing configuration.
- Connect the modem.
- Test the connection.

The following sections explain each step in detail. Make sure you inform the telephone company of your intent to use a modem with your telephone line.



You should be particularly careful, since certain telephone services (such as “call waiting”) can disrupt UUCP conversations.

11.3.1 Choose a Serial Line

Choose the RS-232 serial line you want to use with the system and connect to the modem. If there are no lines available, you must install a new serial line or make one available by removing any device connected to it. If you remove a terminal, make sure no one is logged in.

Once you have chosen a serial line, find the name of the device special file associated with the line by looking in the “XENIX Directories and Special Device Files” chapter of this Guide. The filename should have the form

`/dev/tty nn`

where nn is the number of the corresponding line. For example, `/dev/tty1A` usually corresponds to COM1. You need the name of the actual line for later steps.

Note

`/dev/tty1a` and `/dev/tty1A` correspond to the same physical line; `tty1a` should be used for a direct connection, but `tty1A` should be used for a modem connection.

The serial line you use for your communication system should be owned by `uucp`. To make sure the line is owned by `uucp` enter this command:

`chown uucp /dev/tty nn`

where nn is the number of the corresponding line.

11.3.2 Set the Dialing Configuration

In this communication system, your modem can be used to both send and receive calls. You must set the appropriate switches on the modem. The instructions that follow are specific to Hayes-compatible modems, but other modems are supported. You should refer to the modem manual for connection instructions and see the section “Adding Dial-out Entries to the Devices File” for a complete list of supported modems and dialer programs. (If you are setting up a Hayes Smartmodem 2400 or compatible, see the next section for configuration instructions.) Follow these steps to configure a Hayes Smartmodem 1200 or compatible modem:

1. Remove the front cover of the modem and locate the 8-pin configuration switch. (See the modem reference manual for instructions on how to locate the switch on your particular model.)
2. If the *Devices* file uses the dialer program **dialHA12**, set the pins on the configuration switch to the following positions:

	1	2	3	4	5	6	7	8
up	●	●		●	●	●	●	
down			●					●

If the *Devices* file uses the alternative *Dialers* script `hayes1200`, set the pins as above, but set pin 6 in the “down” position, not “up.”

3. Replace the front cover.

If you have a different modem, consult your reference manual for the proper switch settings to both send and receive calls.

11.3.3 Connect the Modem

Once your modem’s dialing configuration is set, you are ready to connect the modem to your computer. For proper modem operation, the RS-232 cable must provide the pin connections shown below.

Note that the computer’s serial connector must have a DTE (Data Terminal Equipment) configuration. The modem is assumed to have a DCE (Data Communications Equipment) configuration. If this is not the case, you will need to wire the cable from modem to computer with pins 2 and 3 transposed (pin 2 to pin3, and pin 3 to pin 2).

Pin Connections

Name	Computer (DTE)	Modem (DCE)
Protective ground	1	1
Transmit Data (TX)	2	2
Receive Data (RX)	3	3
Data Set Ready (DSR)	6	6
Signal Ground (GND)	7	7
Carrier Detect (CD)	8	8
Data Terminal Ready (DTR)	20	20

These connections are explained in the reference manual for your modem.

Review the installation instructions given in the your modem's manual, then follow these steps:

1. Connect the RS-232 serial cable to the serial line connector on the modem, then to the serial line connector on your computer. Make sure the cable is fully connected. (A 2-3-7 pin cable is not sufficient. We suggest a ribbon cable to connect all appropriate wires.)
2. Plug the telephone line cable into the telephone connector on the modem, then into the telephone wall jack.
3. Plug in the power cord of the modem.

11.3.4 Configuring a Hayes 2400 or Compatible Modem

Although most aspects of modem installation are similar, a Hayes 2400 Smartmodem or compatible modem requires on-line configuration if it is to be used as a dial-in line. Note that the Hayes 2400 will not answer the phone with a 2400 baud carrier if it was not set up with 2400 baud commands.

1. Make sure that the *Devices* file contains an entry for the line:

```
ACU tty $n$ n - 2400 hayes2400
or
ACU tty $n$ n - 2400 /usr/lib/uucp/dialHA24
```



2. You must then configure the modem by issuing set up commands via **cu(C)**. Enter:

```
cu -s2400 -l tty $nn$  dir
```

where *nn* is the “tty” number of the serial line. Press RETURN.

3. Next, enter the following commands to configure the modem. They will be saved in the modem’s non-volatile memory. If you do not want to save the settings, do not enter the last command (**at&w**). Commands are in the left column and short descriptions of what they do are in the right column. Follow each command with a RETURN:

at&f	Fetch factory configuration.
att	Tone dialing.
atl0	Low speaker volume.
at&d2	Set dtr “2”: go on hook when dtr drops.
at&c1	Set dcd “1”: dcd tracks remote carrier.
ats0=1	Answer phone after 1 ring (AA light should come on).
ats2=128	Disable modem escape sequence.
ate0	No echo (modem will no longer echo what is sent to it).
atq1	Quiet mode (modem will not respond with “OK” after this command or any that follow).
at&w	Saves settings in non-volatile memory.

Exit from **cu** by entering a “tilde” and a “period”, followed by RETURN:

(Sometimes it is necessary to press RETURN once before entering the tilde-period.)

The modem is now configured and ready for testing.

11.3.5 Variable Rate Modems



Some modems can determine the connection baud rate from the carrier sent by a remote system. These modems inform the local system of the connection baud rate before issuing the carrier detect signal. The Hayes 2400 dialer supplied with UUCP detects different connection baud rates and informs UUCP and **cu** when it exits with a successful connection.

The speed fields in *Devices* and *Systems* can specify a range of baud rates for a connection. If a dialer supports baud rates from 300 to 2400 baud, enter the baud rate range in the speed field of *Devices* as follows:

```
300-2400
```

If a dialer/modem does not allow variable baud rates, place a single baud in the speed field. If a remote system supports several different speeds, place the range of baud rates in the speed field of *Systems*. If the remote system connects at a single baud rate, place that number in *Systems*. UUCP passes the intersection of the *Systems* and *Devices* baud rate ranges to the dialer when connecting. If the dialer connects outside of the baud range, it returns a bad baud rate error. Otherwise, it returns the baud rate of the connection.

11.3.6 Test the Modem

As the last step of the modem installation, you should test the modem to make sure that it can send and receive calls. Once you have verified that the modem is working, you can begin to use the communication system.

To test the modem, follow these steps:

1. Start the computer and log in as the super user.
2. Disable the modem serial line by entering

```
disable /dev/ttynn
```

where *nn* is the "tty" number of the serial line.

3. Turn on power to the modem.
4. If you are using a Hayes 1200 or compatible, make sure the volume switch on the modem is at an appropriate level. You must be able to hear the modem to carry out this test successfully. Refer to your modem reference manual for the location of this switch.

5. Ensure that the *Systems* file has an entry for the system you intend to call, and that the *Devices* file has a matching entry for *tytnn*.
6. Start the **uucp** program by entering:

/usr/lib/uucp/uucp *sitename*

7. Listen carefully to the modem. You should hear each digit as the number is dialed, then hear the busy signal when the telephone system tries to make connection with your modem.
8. If the busy signal is present, wait a few moments and listen carefully for the modem to hang up. The modem automatically discontinues any call that it cannot make a connection for.
9. If the busy signal is not present, make certain:
 - you have connected the modem to the telephone jack
 - the jack is connected to the phone system
 - you gave the correct phone number in the *Systems* file
10. If you do not hear the modem dial, make certain:
 - the volume switch is up
 - the modem is connected to the correct serial line and that the cable connection is tight
 - you gave the correct *tytnn* line in the *Devices* file
 - modem's power is on.

11.4 Configuring UUCP on Your System

In order to configure your UUCP system, you must edit a series of files which contain information about, and control the actions of the UUCP programs. The UUCP control files are in the **/usr/lib/uucp** directory. You can modify these files with a standard text editor, or use the **uucpinstall(ADM)** program as described below. The descriptions in the latter part of this section provide details on the structure of these files so you can edit them manually.

11.4.1 An Important Consideration: Call or Be Called?



There are three ways to configure a UUCP site:

- As a *dial-in* only site.
- As a *dial-out* only site.
- As a dial-in/out site.

As a dial-in site, other computers call up and log in to your system. They can transfer files and execute certain commands.

As a dial-out site, your computer calls up other computers and logs in. Your computer initiates file transfers to and from the remote machine, as well as local and remote command execution.

Note

The terms *dial-in*, *dial-out* and *call* describe the communication process for both direct wire and modem/telephone sites.

11.4.2 Setting Up the Control Files with `uuinstall`

The rest of this section is concerned with the configuration or control files that act as the UUCP database. The `uuinstall`(ADM) utility provides a simple way to configure these files. Read the rest of the chapter to familiarize yourself with the descriptions of each file and the entries required. The `uuinstall` utility includes a complete series of help files (accessed by pressing ? while in the menus) so it will not be necessary to keep referring to the documentation. When you have some understanding of how each of the control files is used, follow this procedure:

1. Invoke `uuinstall` by logging in as `root` and entering the following command:

`/etc/uuinstall`

△ `sysadmsh` users select: System→Configure→Network→UUCP

The main **uuninstall** menu is displayed:

```
UUCP Administration Utility
=====
1. Display or update site or machine name (/etc/systemid)
2. Display or update list of remote sites (Systems)
3. Display or update direct- or dial-out lines (Devices)
4. Display or update direct- or dial-in lines (/etc/ttys)
5. Check consistency of UUCP files
6. Test connection with remote site
7. Convert old UUCP files to new format

Choose an option (1-7), or enter "q" to quit :
```



Use the **uuninstall** options as follows:

- Give your site a name in the */etc/systemid* file using the “Display or update your sitename” option.
 - Choose the devices to be used for dialing-in or out and enter them in the *Devices* file using the “Display or update dial-in or dial-out devices” option.
 - Identify sites your system will have contact with by creating entries in the *Systems* file with the “Display or update list of remote sites” option.
 - Add the tty lines to be used to the */etc/ttys* file using the “Display or update line connections” option.
2. If other systems will be calling yours, create login accounts using the **mkuser**(ADM) program.
 3. If other systems will be calling yours, define a security scheme that includes what commands and directories can be used in the *Permissions* file.

You will note that some files have many options; where possible, less commonly used options and control files are called out and discussed in the “Special UUCP Configuration Options” section.

When you install the UUCP system, or make any modifications, you should be logged in as super user (root). Virtually all of the UUCP files are writable only by the super user, and many of them are also readable and executable only by **root** and **uucp**. Make sure when you are done that all of the UUCP files are owned by **uucp** and not **root**. UUCP will not work correctly if it cannot read or execute its files.



Note

The files *Systems* and *Permissions* contain unencrypted passwords, and should therefore be readable only by **uucp** (and **root**). Note also that the program */usr/bin/ct* must, exceptionally, be owned by **root** and not by **uucp** in order to work correctly.

11.4.3 Establish Your Site Name in the */etc/systemid* File

In a UUCP system, every computer belongs to a *site*. A site is any computer or any **micnet** network that can communicate with the UUCP system.

To distinguish one site from another, every site must have a unique *sitename*. A *sitename* is any combination of letters and digits that begins with a letter and is no more than seven characters long. The UUCP and **uux** commands use the *sitename* to direct transmissions to the appropriate computer or **micnet** network.

The *sitename* should suggest some characteristic of the site, such as its location or affiliation. For example, a site in Chicago can be named *chicago*, or a site in the legal department can be named *legal*. The *sitename* must be unique. That is, no other computer that calls your computer or is called by your computer can have the same *sitename*.

Each site must have a */etc/systemid* file. The file defines the *sitename* of the given site and associates the site with a **micnet** network, if any. The file has the following form:

```
sitename  
[ machinename ]
```

where:

sitename is the name of the given site.

machinename is the **micnet** machine name for that computer. If the system is not connected to a **micnet** network, the *machinename* is optional.

For example, the following entry defines a site named *chicago* whose **micnet** machine name is *brewster*:

```
chicago
brewster
```



Since UUCP systems are often created after a **micnet** network has been established, the *systemid* file often already exists on a given site. In this case, you must add the *sitename* to the beginning of each *systemid* file on each computer in the **micnet** network.

Note that you can list more than one machine name if desired, but each name must be on a separate line. For a full description, see **systemid(M)** in the *XENIX User's Reference*.

11.4.4 Selecting and Defining a UUCP port

As discussed earlier, you must select a serial line, disable it if it is to be used for dialing-out only or enable it for dialing-in, and edit the serial line entry in the */etc/ttys* file.

1. Select the serial line. Use a line with modem control (for example, */dev/tty1A*) for a dial-in or dial-out line, or a line without modem control (for example, */dev/tty2a*) for a direct connection. For more information, see the section on "Choosing a Serial Line."
2. Disable the serial line. If you are using a modem, be sure it is installed and tested. If the serial line is to be a dial-in line, substitute **enable** for **disable** and enter the command:

```
disable /dev/ttynn
```

where *nn* is the number of your serial line. If the line is already disabled/enabled, the command displays an error message that you can safely ignore.

3. Edit the */etc/ttys* file. This file contains a list of possible login terminals. Enter the following command to display the current entries for the different serial lines:

```
cat /etc/ttys
```

tty entries have the following form:

```
xytty $nn$ 
```

where:

xy is two digits. The first digit is either a one (1), which means the line is enabled; or a zero (0), which means the line is disabled. The second digit is a number or letter that defines the baud rate of the line.

nn is the number of the **tty**.

If you need to change an entry, you can do so with a text editor. For more information on the */etc/ttys* file and the various control codes, see **getty(M)** in the *XENIX User's Reference*.

For example, an entry for a dial-out line (connected to a modem) might look like this:

```
03tty2A
```

In this example, the first digit, 0, means the line is disabled, so that terminals or computers cannot log in on that line. That digit changes to a one when you use the **enable** command. The second digit, 3, means that the **getty** running on that line cycles the baud rate of that line between 1200 and 300 baud. The third digit, 2A, is the number of the serial line.

An example entry for a direct line between two computers might be:

```
06tty2a
```

If a line is to be used for dial-in, ensure that the baud rate code *y* for that line in */etc/ttys* is correct. If the line is to be shared between dial-in and dial-out, ensure that it has an appropriate entry in */usr/lib/uucp/Devices* **and** in */etc/ttys*.

```
enable /dev/ttynn
```

11.4.5 Creating Login Accounts for Sites Dialing-in

A dial-in site must provide a login entry for the sites that call it. These entries are placed in the */etc/passwd* file.

A UUCP login entry has the same form as an ordinary user login entry (see “Preparing XENIX for Users” in this guide), but has a special login directory and login program instead of the normal user directory and shell.

Note

“uucp” should not be used as the name of a uucp user; it is the name of the uucp owner/administrator.

To create a UUCP login entry, use the **mkuser**(ADM) program and follow these steps:

1. Choose a new username and a user ID (identification number) for the UUCP login. The name can be any combination of letters and digits that is no more than eight characters long. The user ID must be an integer in the range 50 to 65535.

Make sure the name and ID are unique. A UUCP login entry must not have the same name or ID as any other login entry.

2. To invoke **mkuser** program, enter:

/etc/mkuser

Follow the program menus and prompts to add the account(s) you wish.

3. The **mkuser** program prompts you to enter a password for the new user. Passwords are optional for UUCP logins.
4. Choose option 5 (uucp) as the shell for a UUCP user; this gives uucico as a shell, and a home directory in */usr/spool/uucp*.

11.4.6 Adding Entries for Remote Sites to the Systems File



The *Systems* file (*/usr/lib/uucp/Systems*) contains the information needed by the **uucico** daemon to establish a communication link to a remote computer. Each entry in the file represents a computer that can be called by your computer. In addition, the *Systems* file can be configured to prevent any computer that does not appear in this file from logging in on your computer. More than one entry may be present for a particular computer. The additional entries represent alternative communication paths that will be tried in sequential order.

Note

If you are setting up your system as a *dial-in* only (passive) site that never initiates calls, you only need add the names of the systems that will be calling you.

Each entry in the *Systems* file has the following format:

sitename schedule device speed phone login-script

- | | |
|---------------------|---|
| <i>sitename</i> | field contains the node name of the remote computer. |
| <i>schedule</i> | field is a string that indicates the day-of-week and time-of-day when the remote computer can be called. |
| <i>device</i> | is the device type that should be used to establish the communication link to the remote computer. |
| <i>speed</i> | indicates the transfer speed of the device used in establishing the communication link. |
| <i>phone</i> | provides the phone number of the remote computer for automatic dialers. If you wish to create a portable <i>Systems</i> file that can be used at a number of sites where the dialing prefixes differ (for internal phone systems), refer to "Using Dialcodes to Create a Portable Systems File" under "Special UUCP Configuration Options." |
| <i>login-script</i> | contains login information (also known as a "chat script"). |

The Schedule Field

The *schedule* consists of three subfields. The first, *day*, is required. The other two, *time* and *retry*, are optional. The syntax is as follows:

day[*time*][:*retry*]

The *day* subfield can contain the following keywords:

<i>Su Mo Tu We Th Fr Sa</i>	For individual days.
<i>Wk</i>	For any weekday (Mo Tu We Th Fr).
<i>Any</i>	For any day.
<i>Never</i>	For a passive arrangement with the remote computer. If the <i>schedule</i> field is Never , your computer will never initiate a call to the remote computer. The call must be initiated by the remote computer. In other words, your computer is in a passive mode in respect to the remote computer (see discussion of <i>Permissions</i> file).

The optional *time* subfield should be a range of times in 24-hour clock format, such as 0800-1230. If no *time* is specified, any time of day is assumed to be allowed for the call. A time range that spans 0000 is permitted. For example, **0800-0600** means all times are allowed other than times between 6 AM and 8 AM. *sp .5* For example, the following permits calls on Mondays, Wednesdays, and Fridays between the hours of 9 AM and Noon (the *schedule* field is in boldface for clarity):

```
grebe MoWeFr0900-1200 ACU,g D1200 14087672676 \  
ogin: nuucp ssword: Crested
```

You can also specify more than one set of *day* and *time* entries. This is useful for more complex specifications. The following example allows calls from 5:00 p.m. to 8:00 am, Monday through Thursday, and calls any time Saturday and Sunday. The example would be an effective way to call only when phone rates are low, if immediate transfer is not critical:

```
gorgon Wk1700-1800, SaSu ACU,g D1200 14087672676 \  
ogin: nuucp ssword: DontLook
```

The optional subfield, *retry*, is available to specify the minimum time (in minutes) before a retry, following a failed attempt. The subfield separator is a semicolon (;). For example, the following is interpreted as call any time, but wait at least 9 minutes before retrying after a failure occurs:

Any;9

Note

By default UUCP uses an ‘exponential backoff’ method to retry failed calls. After the initial failure, a second call is made in five minutes. This interval expands as the number of unsuccessful attempts increases. The *retry* field is used to override the default.

The Device Field


The *device* field selects the device type, in most cases an ACU (Automatic Calling Unit). For example, the keyword used in the following field is matched against the first field of *Devices* file entries:

```
Systems:  gorgon Any ACU,g D1200 14087672676 \  
          ogin: nuucp ssword: DontLook  
  
Devices:  ACU tty11 - D1200 hayes
```

The Speed Field

This field can contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers (refer to the discussion on the *Devices* file, *speed* field). Some devices can be used at any speed, so the keyword **Any** may be used. However, we recommend that you specify the actual

range of speeds that can be used. (If **Any** is used in both *Systems* and *Devices* entries, 1200 is assumed.) For example, this field must match the *speed* field in the associated *Devices* file entry:



```
Systems:  gorgon Any ACU D2400-9600 14087672676 \  
          ogin: nuucp ssword: DontLook  
Devices:  ACU tty11 - D1200-2400 hayes2400
```

If information is not required for this field, use a hyphen (-) as a place holder for the field.

The Phone Field


This field is used to provide the phone number used for the modem dialer. The phone number is made up of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be one that is listed in the *Dialcodes* file. (See the “Using Dialcodes to Create a Portable Systems File” section for details.) For example:

```
Systems:  gorgon Any ACU D1200 CA2676 \  
          ogin: nuucp ssword: DontLook  
Dialcodes: CA 9=408767
```

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.

If your computer is connected to a LAN switch, you may access other computers that are connected to that switch. The *Systems* file entries for these computers will not have a phone number in the *phone* field. Instead, this field will contain the token that must be passed on to the switch so it will know which computer your computer wishes to communicate with. (This is usually just the system name.) The associated *Devices* file entry should have a **\D** at the end of the entry to ensure that this field is not translated using the *Dialcodes* file.

The Login-Script Field



The login-script is used to open communications between modems, plus recognize and send proper login and password sequences. The script is given as a series of fields and subfields of the following format:

expect send

where *expect* is the string that is received, and *send* is the string that is sent when the *expect* string is received.

The *expect* field can be made up of subfields of the following form:

expect[-subsend-subexpect]. ..

where the *subsend* is sent if the prior *expect* is not successfully read and the *subexpect* following the *subsend* is the next expected string. To make this distinction clear: the send-expect sequence sends a string if the expect string is received, the subsend-subexpect sends only if the prior expect string is not received.

For example, with "login--login", the UUCP program will expect "login". If a "login" is received, it will go on to the next field. If it does not get "login", it will send nothing followed by a carriage return, then look for "login" again. If no characters are initially expected from the remote computer, the characters " " (null string) should be used in the first *expect* field. Note that all *send* fields will be sent followed by a carriage return unless the *send* string is terminated with a \c.

The *expect* string need not be complete; only the trailing characters must be specified, as in "ogin:". This avoids difficulties with login strings that use an uppercase letter as in "Login:" or "Password:", and also difficulties when the line is shared by dial-in and dial-out.

Creating Login Scripts

This section explains in greater detail how to create a login (chat) script.

Consider the following sample *Systems* file entry:

```
terps Any ACU 1200 18005211980 "" \r ogin:-BREAK-ogin: \  
uucpx word: ichore
```

This is how this script would work during connection:


1. Nothing is expected initially.
2. A carriage return is sent and the script waits for the prompt “ogin:” (login:).
3. If it doesn’t receive “ogin:”, send a BREAK.
4. When “ogin:” is finally received, send the login name **uucpx**.
5. When the prompt, “word:” (for Password:) is received, send the password “ichore”.



Login (chat) scripts will often require some experimentation. There will be cases where a modem will require one or more BREAK sequences before presenting a login (this is often true with variable speed modems). If you cannot obtain the necessary login sequence from the system administrator for a given site, it is a good idea to connect with the site manually. You can accomplish this using **cu** and find out what must be sent to generate a login prompt. (You can also connect with a system using a verbose option for debugging; see “Debugging Transmissions” for details.) There are several escape characters that cause specific actions when sent during the login sequence, some of which correspond to keystrokes; these should be included in the script where necessary:

Login (Chat) Script Escape Sequences

Character	Description
\N	Sends or expects a null character (ASCII NUL).
\b	Sends or expects a backspace character.
\c	If at the end of a string, suppresses the carriage return that is normally sent. Ignored otherwise.
\d	Delays two seconds before sending or reading more characters.
\p	Pauses for approximately ¼ to ½ second.
\E	Starts echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.)
\e	Turns echo check off.
\n	Sends or expects a new-line character.



<code>\r</code>	Sends or expects a carriage-return.
<code>\s</code>	Sends or expects a space character.
<code>\t</code>	Sends or expects a tab character.
<code>\\</code>	Sends or expects a <code>\</code> character.
<code>EOT</code>	Sends or expects EOT new-line twice.
<code>BREAK</code>	Sends or expects a break character.
<code>\K</code>	Same as <code>BREAK</code> .
<code>\ddd</code>	Collapses the octal digits (<code>ddd</code>) into a single character.

11.4.7 Limiting Access with the Permissions File

If other machines will be dialing into your system, the *Permissions* file (*/usr/lib/uucp/Permissions*) specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the remote computer's ability to request files and its ability to receive files queued by the local site. Other options specify the commands that a remote site can execute on the local computer.

Structuring Permissions File Entries

Each entry is a logical line with physical lines terminated by a `\` to indicate continuation. Entries are made up of options delimited by white space. Each option is a name-value pair in the following format:

name=value

Note that no white space is allowed within an option assignment.

Comment lines begin with a pound sign (`#`) and they occupy the entire line up to a newline character. Blank lines are ignored (even within multi-line entries).

There are two types of *Permissions* file entries:

- | | |
|----------------------|---|
| <code>LOGNAME</code> | Specifies the permissions that take effect when a remote computer logs in on (calls) your computer. |
| <code>MACHINE</code> | Specifies permissions that take effect when your computer logs in on (calls) a remote computer. |

Considerations on Permissions File Restrictions

The following should be considered when using the *Permissions* file to restrict the level of access granted to remote computers:



- All login IDs used by remote computers to login for UUCP communications must appear in only one LOGNAME entry.
- Any site that is called whose name does not appear in a MACHINE entry, will have the following default permissions/restrictions:
 1. Local send and receive requests will be executed.
 2. The remote computer can send files to your computer's */usr/spool/uucppublic* directory.
 3. The commands sent by the remote computer for execution on your computer must be one of the default commands, usually **rmail**.

Permissions Options

This section describes each option, specifies how they are used, and lists their default values.

REQUEST

Specifies whether the remote computer can request to set up file transfers from your computer. When a remote computer calls your computer and requests to receive a file, this request can be granted or denied. The following string specifies that the remote computer can request to transfer files from your computer:


```
REQUEST=yes
```

The following string specifies that the remote computer cannot request to receive files from your computer:

```
REQUEST=no
```

The *no* value is the default value. It will be used if the REQUEST option is not specified. The REQUEST option can appear in either a LOGNAME (remote calls you) entry or a MACHINE (you call remote) entry. A note on security: When a remote machine calls you, unless you have a unique login and password for that machine you don't know if the machine is who it says it is.

SENDFILES



Specifies whether your computer can send the work queued for the remote computer. When a remote computer calls your computer and completes its work, it may attempt to take work your computer has queued for it.

The following string specifies that your computer can send the work that is queued for the remote computer as long as it logged in as one of the names in the LOGNAME option:

```
SENDFILES=yes
```

This string is mandatory if your computer is in a *passive mode* with respect to the remote computer.

The following string specifies that files queued in your computer be sent only when your computer calls the remote computer:

```
SENDFILES=call
```

The *call* value is the default for the SENDFILE option. This option is only significant in LOGNAME entries since MACHINE entries apply when calls are made out to remote computers. If this option is used with a MACHINE entry, it will be ignored.

READ and WRITE

Specify the various parts of the file system that **uucico** can read from or write to. The READ and WRITE options can be used with either MACHINE or LOGNAME entries.

The default for both the READ and WRITE options is the *uucpublic* directory as shown in the following strings:

```
READ=/usr/spool/uucpublic  
WRITE=/usr/spool/uucpublic
```

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 the prefix of any full path name of a file coming in or going out.

Note

READ and WRITE options do not effect the actual permissions of a file or directory. For example, a directory with permissions of 700 will only permit the owner to access it, no matter what access options are defined in the *Permissions* file.



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=/usr/spool/uucppublic:/usr/news
```

It should be pointed out that if the READ and WRITE options are used, all path names must be specified because the path names are not added to the default list. For instance, if the */usr/news* path name was the only one specified in a WRITE option, permission to deposit files in the public directory would be denied.

You should be careful what directories you make accessible for reading and writing by remote systems. For example, you probably wouldn't want remote computers to be able to write over your */etc/passwd* file so */etc* shouldn't be open to writes.


NOREAD and NOWRITE

Specify exceptions to the READ and WRITE options or defaults. The following strings would permit reading any file except those in the */etc* directory (and its subdirectories—remember, these are prefixes) and writing only to the default */usr/spool/uucppublic* directory:

```
READ=/
WRITE=/usr/spool/uucppublic
NOREAD=/etc
NOWRITE=/etc
```

NOWRITE works in the same manner as the NOREAD option. The NOREAD and NOWRITE can be used in both LOGNAME and MACHINE entries.

CALLBACK



Specifies in LOGNAME entries that no transaction will take place until the calling system is called back. There are two examples of when you would use CALLBACK. >From a security standpoint, if you call back a machine you can be sure 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 computer must call the remote computer back before any file transfers will take place:

```
CALLBACK=yes
```

The default for the COMMAND option is

```
CALLBACK=no
```

The CALLBACK option is very rarely used. Note that if two sites have this option set for each other, a conversation will never get started.

COMMANDS

Specifies the commands in MACHINE entries that a remote computer can execute on your computer. This affects the security of your system; use it with extreme care.

The **uux** program will generate remote execution requests and queue them to be transferred to the remote computer. Files and a command are sent to the target computer for remote execution. Note that COMMANDS is not used in a LOGNAME entry; COMMANDS in MACHINE entries define command permissions whether you call the remote system or it calls you.

The following string indicates the default commands that a remote computer can execute on your computer:

```
COMMANDS=rmail
```

If a command string is used in a MACHINE entry, the default commands are overridden. For instance, the following entry overrides the COMMAND default so that the computers *owl*, *raven*, *hawk*, and *dove* can now execute **rmail**, **rnews**, and **lp** on your computer:

```
MACHINE=owl:raven:hawk:dove \  
COMMANDS=rmail:rnews:lp
```

Building a Remote Network with UUCP

In addition to the names as specified above, there can be full path names of commands. For example, the following command specifies that command **rmail** uses the default path:

```
COMMANDS=rmail:/usr/sbin/rnews:/usr/local/lp
```

The default paths for your computer are */bin*, */usr/bin* and */usr/lbin*. When the remote computer specifies **lp**, **/usr/local/lp** is the command that will be executed.

Including the *ALL* value in the list means that any command from the remote computer specified in the entry will be executed. If you use this value, you give the remote computer full access to your computer. So, be careful; this allows far more access than normal users have.

The following string illustrates two points:

```
COMMANDS=/usr/local/bin/lc:ALL:/usr/local/lp
```

1. The *ALL* value can appear anywhere in the string; and the path names specified for **lc** and **lp** will be used (instead of the default) if the requested command does not contain the full path names for **lc** or **lp**.
2. The *VALIDATE* option should be used with the *COMMANDS* option whenever potentially dangerous commands like **cat** and **uucp** are specified with the *COMMANDS* option. Any command that reads or writes files is potentially dangerous to local security when executed by the UUCP remote execution daemon (**uuxqt**).

VALIDATE

Used in conjunction with the *COMMANDS* option when specifying commands that are potentially dangerous to your computer's security. It provides a certain degree of verification of the caller's identity. The use of the *VALIDATE* option requires that privileged computers have a unique login/password for UUCP transactions. An important aspect of this validation is that the login/password associated with this entry be protected. If an outsider gets that information, that particular *VALIDATE* option can no longer be considered secure. (*VALIDATE* is merely an added level of security to the *COMMANDS* option, though it is a more secure way to open command access than *ALL*.)





Careful consideration should be given to providing a remote computer with a privileged login and password for UUCP transactions. Giving a remote computer a special login and password with file access and remote execution capability is like giving anyone on that computer a normal login and password on your computer. Therefore, if you cannot trust someone on the remote computer, do not provide that computer with a privileged login and password.

The following LOGNAME entry specifies that if one of the remote computers that claims to be *eagle*, *owl*, or *hawk* logs in on your computer, it must have used the login **uucpfriend**:

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

As can be seen, if an outsider gets the **uucpfriend** login/password, masquerading is trivial.

COMMAND options appearing in MACHINE entries define the commands available to the system, and to the machine's LOGNAME entry. Commands need to be defined in MACHINE entries in order to continue communication between the local and remote systems when the remote computer is logged on to the local system and the execution daemon is not running.

Each remote computer has its own *spool* directory on your computer. These spool directories have write permission given only to the UUCP programs. The execution files from the remote computer are put in its spool directory after being transferred to your computer. When the **uuxqt** daemon runs, it can use the spool directory name to find the MACHINE entry in the *Permissions* file and get the COMMANDS list. Or, if the computer name does not appear in the *Permissions* file, the default list will be used.

The following example shows the relationship between the MACHINE and LOGNAME entries:

```
MACHINE=eagle:owl:hawk REQUEST=yes \  
COMMANDS=rmail:/usr/local/bin/lc \  
READ=/ WRITE=  
  
LOGNAME=uucpz VALIDATE=eagle:owl:hawk \  
REQUEST=yes SENDFILES=yes \  
READ=/ WRITE=
```

The value in the COMMANDS option means that remote mail and **/usr/local/bin/lc** can be executed by remote users.

In the first entry, you must make the assumption that when you want to call one of the computers listed, you are really calling either *eagle*, *owl*, or *hawk*. Therefore, any files put into one of the *eagle*, *owl*, or *hawk* spool directories is put there by one of those computers. If a remote computer logs in and says that it is one of these three computers, its execution files will also be put in the privileged spool directory. You therefore have to validate that the computer has the privileged login **uucpz**.



MACHINE Entry for OTHER Systems

You may want to specify different option values for the computers your computer calls that are not mentioned in specific MACHINE entries. This may occur when there are many computers calling in, and the command set changes from time to time. The name *OTHER* for the computer name is used for this entry as follows:

```
MACHINE=OTHER \  
COMMANDS=rmail:lc:/usr/local/bin/lc
```

All other options available for the MACHINE entry can also be set for the computers that are not mentioned in other MACHINE entries.

Combining MACHINE and LOGNAME Entries

It is possible to combine MACHINE and LOGNAME entries into a single entry where the common options are the same. For example, the following two 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=
```

These two entries can be merged as follows:

```
MACHINE=eagle:owl:hawk REQUEST=yes \  
LOGNAME=uucpz SENDFILES=yes \  
READ=/ WRITE=
```

11.4.8 Adding Dial-out Entries to the the Devices File



The *Devices* file (*/usr/lib/uucp/Devices*) contains information for all the devices that can be used to establish a link to a remote computer, devices such as automatic call units, direct links, and network connections. This file works closely with the *Dialers*, *Systems*, and *Dialcodes* files. Before you make changes in any of these files, you should be familiar with them all. Note that a change to an entry in one file may require a change to a related entry in another file.

Each entry in the *Devices* file has the following format:

```
type ttyline dialerline speed dialer-token
```

where:

- | | |
|---------------------|--|
| <i>type</i> | can contain one of two keywords (direct or ACU), the name of a Local Area Network switch, or a system name. |
| <i>ttyline</i> | contains the device name of the line (port) associated with the <i>Devices</i> entry. For example, if the Automatic Dial Modem for a particular entry was attached to the <i>/dev/tty11</i> line, the name entered in this field would be tty11 . |
| <i>dialerline</i> | This option is useful only for 801 type dialers, which do not contain a modem and must use an additional line. Unless you have an 801 dialer, simply enter a hyphen (-) as a placeholder. |
| <i>speed</i> | is the speed or speed range of the device. Can also contain an indicator for distinguishing different dialer classes. |
| <i>dialer-token</i> | This field contains pairs of dialers and tokens, each representing a dialer and an argument to be passed to it. The <i>dialer</i> portion can be the name of an automatic dial modem, or it may be a direct for a direct link device. |

The Type Field

This field can contain one of two keywords (**direct** or **ACU**), the name of a Local Area Network switch, or a system name:



- Direct** This keyword indicates a direct link to another computer or a switch (for **cu** connections only).
- ACU** This keyword indicates that the link to a remote computer is made through an automatic call unit (Automatic Dial Modem). This modem can be connected either directly to your computer or indirectly through a Local Area Network (LAN) switch.
- LANswitch** can be replaced by the name of a LAN switch. **micom** and **develcon** are the only ones that have caller scripts in the *Dialers* file.
- sysname** indicates a direct link to a particular computer. (*sysname* is replaced by the name of the computer.) This means that the line associated with this *Devices* entry is for a particular computer in the *Systems* file.

For example the keyword “gorgon” used in the *Type* field *Devices* is matched against the third field of *Systems* file entries:

```
Devices: gorgon tty11 - 1200 hayes1200
Systems: gorgon Any gorgon 1200 14087672676 ogin: nuucp \
        ssword: DontLook
```

The Speed Field

In most cases, this is simply the speed of the device, if the keyword **ACU** or **Direct** is used in the *type* field. However, *speed* may contain a letter and a speed (for example, C1200, D1200) to differentiate between classes of dialers (Centrex or Dimension PBX). This is necessary because many larger offices may have more than one type of telephone network: one network may be dedicated to serving only internal office communications, while another handles the external communications. In such a case, it becomes necessary

to distinguish which lines should be used for internal communications and which should be used for external communications. The keyword used in the *speed* field of the *Devices* file is matched against the fourth field of *Systems* file entries, for example:

```
Devices:  ACU tty11 - D1200 hayes1200
Systems:  gorgon Any ACU D1200 3251 ogin: nuucp \
          ssword: DontLook
```

Some devices can be used at any speed, so the keyword **Any** can be used in the *speed* field. If **Any** is used, the line will match any speed requested in a *Systems* file entry. If this field is **Any** and the *Systems* file *speed* field is **Any**, the speed defaults to 1200 bps. If a device can be used at a range of speeds, then the speed field can specify this range (for example, 1200-9600 or D1200-9600). This is preferable to the use of **Any**.

The Dialer-Token Field

This field has the following format:

dialer [token dialer token ...]

For a direct line, this field contains simply the word **direct**, and no token is required.

For a simple connection to a dialer, this field contains the name of the dialer, and the token is omitted; by default it is taken from the phone number field of the *Systems* file entry.

For a dialer or a network dataswitch, this field contains the name of an entry found in the *Dialers* file (**develcon** and **micom** are examples of network data switches). Other dialer types are supported by binaries instead of *Dialers* entries. (Support for 801-type dialers is provided through use of separate lines for data and the dialer. See the *Devices* file for details.) UUCP recognizes a dialer as a binary if the name begins with a "/" or there is an executable file by that name in */usr/lib/uucp*.

The following dialer types are available as *Dialers* entries:

Dialer type	Modem or Data Switch
Direct	Direct line; no dialer
Penril	Penril modem
Hayes	Hayes modem (or compatible)
Ventel	Ventel 212+ modem
Vadic	Racal Vadic 3451 modem
LANswitch	Network switch described in type field
Hayes1200	Hayes Smartmodem 1200
Hayes2400	Hayes Smartmodem 2400
Develcon	Develcon network dataswitch
Micom	Micom network dataswitch
Rixon	Rixon Intelligent Modem
ATT4000	AT&T Programmable 300/1200 Modem Model 4000
ATT2212c	AT&T DATAPHONE II 2212C Modem
ATT2224	AT&T DATAPHONE II 2224 Modem
NLS	Network Listener Service



Note

The TLI and TLIS dialer types are currently not available with XENIX.

The following binary types are provided in *usr/lib/uucp*:

Binary file	Modem
dialHA12	Hayes Smartmodem 1200 or compatible
dialHA24	Hayes Smartmodem 2400 or compatible
dialVA3450	Racal Vadic 3451 modem
dialTBIT	Telebit Trailblazer Modem

(The source is also provided for these dialer binaries; you can adapt and compile your own dialers if desired. See the section “Dialing Out from Your Computer” in the “Using Terminals and Modems” chapter of this guide for details.)

Structuring Dialer-Token Entries

The *dialer-token* can be structured four different ways, depending on the device associated with the entry:

1. **Simple modem connection.** If an automatic dialing modem is connected directly to a port on your computer, the *dialer-token* field of the associated *Devices* file entry will only have one pair. This pair would normally be the name of the modem. This name is used to match the particular *Devices* file entry with an entry in the *Dialers* file. Therefore, the *dialer* field must match the first field of the following *Dialers* file entry:

```
Devices:   ACU tty11 - 1200 ventel
Dialers:   ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

Notice that only the *dialer* portion (**ventel**) is present in the *dialer-token* field of the *Devices* file entry. This means that the *token* to be passed on to the dialer (in this case the phone number) is taken from the *Phone* field of a *Systems* file entry. (\T is implied, see item 4 below.) Backslash sequences are described later.

2. **Direct links.** If a direct link is established to a particular computer, the *dialer-token* field of the associated entry would contain the keyword **direct**. This is true for both types of direct link entries, **direct** and *sysname* (refer to discussion on the *type* field).
3. **Local network switches.** If a computer that you wish to communicate with is on the same local network switch as your computer, your computer must first access the switch and the switch can make the connection to the other computer. In this type of entry, there is only one pair. The *dialer* portion is used to match a *Dialers* file entry following:

```
Devices:   develcon tty13 - 1200 develcon \D
Dialers:   develcon "" "" \pr\ps\c est:\007 \E\D\e \007
```

As shown, the *token* portion is **\D**, which indicates that it is retrieved from the *Systems* file without translation. The *Systems* file entry for this particular computer will contain the token in the *phone* field; this

is normally reserved for the phone number of the computer (refer to *Systems* file, *phone* field). The **\D** ensures that the contents of the *phone* field will not be interpreted as a valid entry in the *Dialcodes* file.

4. **Modems used with a local network switch.** If an automatic dialing modem is connected to a switch, your computer must first access the switch and the switch will make the connection to the automatic dialing modem. This type of entry requires two *dialer-token-pairs*. The following *dialer* portion of each pair (fifth and seventh fields of entry) will be used to match entries in the *Dialers* file:

```
Devices:   ACU tty14 - 1200 develcon vent ventel
Dialers:   develcon "" "" \pr\ps\c est:\007 \E\D\e \007
           ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

In the first pair, **develcon** is the switch and **vent** is the token that is passed to the **develcon** switch to tell it which device to connect to your computer. This token would be unique for each LAN switch since each switch may be set up differently. Once the ventel modem has been connected, the second pair is accessed, where ventel is the dialer and the token is retrieved from the *Systems* file.

The following are two escape characters that can appear in the *dialer-token* field:

- \T** indicates that the *Phone* field should be translated at this stage, using the *Dialcodes* file. This escape character is normally placed in the *Dialers* file for each caller script associated with an automatic dial modem (penril, ventel, etc.). The translation will not take place until the caller script is accessed.
- \D** indicates that the *Phone* field should not be translated using the *Dialcodes* file. If no escape character is specified at the end of a *Devices* entry, **\D** is assumed by default when a *Dialers* script is to be used (which can itself contain a **\T** to translate the number). **\T** is assumed if a built-in or dialer binary is to be used (because there is then no later opportunity to translate the number).

11.4.9 Using the Same Port for Dialing In and Out

11

It is possible to dial in and out on the same line without enabling/disabling the line or running a special version of **getty**. All that is necessary is to first create an entry for a line in the *Devices* file (dial-out) and then an entry in *letcttys* (dial-in) for the same line. When access to a dial-out line is requested on a shared port, **getty** runs a special program, **uuchat**, that automatically reinitializes the port when the call is complete. **uuchat** uses special dialer scripts found in the *Dialers* file that begin with an ampersand. This means there are actually two entries for some dialers. For example, the dialer for the Hayes Smartmodem 2400 (or compatible) consists of two entries: **hayes2400** and **&hayes2400**, the latter of which is used when reinitializing a shared port to dial-in. In the case of the dialer binaries in */usr/lib/uucp*, these programs are automatically invoked with the **-h** (hangup) switch that reinitializes the port to dial-in.

11.5 Special UUCP Configuration Options

This section contains several options that are used for special circumstances and can be ignored in most cases.

11.5.1 Adding Dialers to the Dialers File

The *Dialers* file (*/usr/lib/uucp/Dialers*) specifies the initial conversation that must take place on a line before it can be made available for transferring data. This conversation is usually a sequence of ASCII strings that is transmitted and expected, and it is often used to dial a phone number using an ASCII dialer (such as the Automatic Dial Modem).

A modem that is used for dialing in and out may require a second *Dialers* entry. This is to reinitialize the line to dial-in after it has been used for dial-out. The name of the dial-in version of a dialer must begin with an ampersand. For example, the *Dialers* file contains a **hayes2400** and a **&hayes2400** entry.

As shown in the earlier examples, the fifth field in a **Devices** file entry is an index into the **Dialers** file or a special dialer type (801, for example). Here an attempt is made to match the fifth field in the *Devices* file with the first field of each *Dialers* file entry. In addition, each odd numbered *Devices* field starting with the seventh position is used as an index into the *Dialers* file. If the match succeeds, the *Dialers* entry is interpreted to perform the dialer negotiations. Each entry in the *Dialers* file has the following format:

dialer substitutions expect-send ...

The *dialer* field matches the fifth and additional odd numbered fields in the *Devices* file. The *substitutions* field is a translate string: the first of each pair of characters is mapped to the second character in the pair. This is usually used to translate = and - into whatever the dialer requires for “wait for dial-tone” and “pause.”

The remaining *expect-send* fields are character strings. Below are some character strings distributed with the UUCP package in the *Dialers* file.

<i>Dialers</i> file entries	
penril	=W-P "" \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!
hayes	=,-, "" \dAT\r\c OK\r \EATDT\T\r\c CONNECT
rixon	=&-% "" \d\r\r\c \$ s9\c)-W\r\ds9\c-) s\c : \T\r\c \$ 9\c LINE
vadiac	=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 \007
micom	"" "" \s\c NAME? \D\r\c GO
direct	
att2212c	=+,-, "" \r\c :--: ato12=y,T\T\r\c red
att4000	=,-, "" \033\r\r\c DEM: \033s0401\c \006 \033s0901\c \006 \006 \033s1001\c \006 \033s1102\c \006 \033dT\T\r\c \006
att2224	=+,-, "" \r\c :--: T\T\r\c red
nls	"" "" NLPS:000:001:1\N\c

The meaning of some of the escape characters (those beginning with “\”) used in the *Dialers* file are listed below:

\p	pause (approximately ¼ to ½ second)
\d	delay (approximately 2 seconds)
\D	phone number or token without <i>Dialcodes</i> translation
\T	phone number or token with <i>Dialcodes</i> translation
\K	insert a BREAK
\E	enable echo checking (for slow devices)
\e	disable echo checking
\r	carriage return
\c	no new-line or carriage return
\n	send new-line
\nnn	send octal number.

Additional escape characters that may be used are listed in the section discussing the *Systems* file.

The penril entry in the *Dialers* file is executed as follows. First, the phone 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 next thing.)
\d	Delay for 2 seconds.
>	Wait for a >.
s\p9\c	Send an s, pause for ½ second, send a 9, send no terminating new-line
)-W\p\r\ds\p9\c-)	Wait for a). If it is not received, process the string between the - characters as follows. Send a W , pause, send a carriage-return, delay, send an s, pause, send a 9, without a new-line, and then wait for the).
y\c	Send a y.
:	Wait for a :.
\E\TP	Enable echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.) Then, send the phone number. The T means take the phone number passed as an argument and apply the <i>Dialcodes</i> translation and the modem function translation specified by field 2 of this entry. Then send a P .
>	Wait for a >.
9\c	Send a 9 without a new-line.
OK	Waiting for the string OK .

11.5.2 Using Dialcodes to Create a Portable Systems File

The *Dialcodes* file (*/usr/lib/uucp/Dialcodes*) contains the dial-code abbreviations that can be used in the *Phone* field of the *Systems* file. This feature is intended primarily for those who wish to create a standard *Systems* file for distribution among several sites that have different phone systems and area codes. As such, the *Dialcodes* file is probably not necessary for most sites.

Dial codes are used to phone system-specific parts of a dial string from the actual phone number. For example, if two remote sites in a network have the same sites to link with, but have different internal phone systems (one has to dial "9" and wait for a dial tone to dial outside the facility and the other does not) each site can share the same *Systems* file, but have different entries in a *Dialcodes* file. Each entry has the following format:



abb dial-seq

where:

abb is the abbreviation used in the *Systems* file *phone* field

dial-seq is the dial sequence that is passed to the dialer when that particular *Systems* file entry is accessed.

The following entry would be set up to work with a *phone* field in the *Systems* file such as *jt7867*:

```
jt 9=847-
```

When the entry containing *jt7867* is encountered, the following sequence would be sent to the dialer if the token in the dialer-token-pair is \T:

```
9=847-7867
```

The phone number is made up of an optional alphabetic abbreviation and a numeric part. For example, if an abbreviation is used, it must be one that is listed in the *Dialcodes* file.

<pre><i>Systems:</i> eagle Any ACU D1200 NY3251 ogin: nuucp \ ssword: Oakgrass</pre>
<pre><i>Dialcodes:</i> NY 9=1212555</pre>

In this string, an equal sign (=) tells the ACU to wait for a secondary dial tone before dialing the remaining digits. A hyphen in the string instructs the ACU to pause before dialing the next digit. The number of seconds may vary from dialer to dialer (for example, 2 seconds for Hayes, 5 seconds for Vadic).

11.5.3 Using a Streams-based Network: the Devconfig File

The `/usr/lib/uucp/Devconfig` file is used when your computer communicates over a Streams-based transport provider that conforms to the AT&T Transport Interface (TI).

Note

STREAMS support requires installation of the XENIX STREAMS Toolkit; it is not included with the XENIX Operating System distribution.

Devconfig entries define the STREAMS modules that are used for a particular TI device. Entries in the *Devconfig* file have the format:

```
service=x device=y push=z[:z...]
```

where *x* can be **cu**, **uucico**, or both separated by a colon; *y* is the name of a TI network and must match an entry in the *Devices* file; and *z* is replaced by the names of Streams modules in the order that they are to be pushed onto the Stream. Different modules and devices can be defined for **cu** and **uucp** services.

The following entries should most commonly be used in the file:

```
service=cu device=STARLAN push=ntty:tirdwr:ld0
service=uucico device=STARLAN push=ntty:tirdwr:ld0
```

This example pushes *ntty*, then *tirdwr*, then *ld0*. The *Devconfig* file cannot be modified with the **uinstall** utility. If you want to change the contents of this file, you must use one of the XENIX text editors, such as **vi(C)**.

11.5.4 Creating Alternate Control Files with Sysfiles

The `/usr/lib/uucp/Sysfiles` file lets you assign different files to be used by **uucp** and **cu** as *Systems*, *Devices*, and *Dialers* files. Here are some cases where this optional file may be useful.

- You may want different *Systems* files so requests for login services can be made to different addresses than UUCP services.
- You may want different *Dialers* files to use different handshaking for **cu** and **uucp**.

- You may want to have multiple *Systems*, *Dialers*, and *Devices* files. The *Systems* file in particular may become large, making it more convenient to split it into several smaller files.



The format of the *Sysfiles* file is

```
service=w systems=x:x dialers=y:y devices=z:z
```

where *w* is replaced by **uucico**, **cu**, or both separated by a colon; *x* is one or more files to be used as the *Systems* file, with each file name separated by a colon and read in the order presented; *y* is one or more files to be used as the *Dialers* file; and *z* is one or more files to be used as the *Devices* file. Each file is assumed to be relative to the */usr/lib/uucp* directory, unless a full path is given. A backslash-carriage return (`\<CR>`) can be used to continue an entry on to the next line.

Here's an example of using a local *Systems* file in addition to the usual *Systems* file:

```
service=uucico:cu systems=Systems:Local_Systems
```

If this is in */usr/lib/uucp/Sysfiles*, then both **uucico** and **cu** will first look in */usr/lib/uucp/Systems*. If the system they're trying to call doesn't have an entry in that file, or if the entries in the file fail, then they'll look in */usr/lib/uucp/Local_Systems*.

When different *Systems* files are defined for **uucico** and **cu** services, your machine will store two different lists of *Systems*. You can print the **uucico** list using the *uname* command or the *cu* list using the *uname -c* command.

11.5.5 Preventing Unknown Sites from Logging in

The script **remote.unknown** is executed when a site whose name is not recognized dials in to your system. It will log the conversation attempt and fail to make a connection. If you wish to allow such "unknown" systems to log in to your system, you can change the permissions of this file so it cannot execute and your system will accept any communication requests. To do so, enter the following commands while logged-in as **root**:

```
cd /usr/lib/uucp
chmod 000 remote.unknown
```

11.5.6 Linking Micnet Sites with the UUCP Network



To use a UUCP system with your **micnet** network, follow these steps:

1. Add the following entry to the *aliases* file of the computer on which the UUCP system is installed:

```
uucp:
```

2. For all other computers in your site, add the following entry to the *aliases* file:

```
uucp:machine-name:
```

where *machine-name* is the name of the computer on which the UUCP system is installed. This longer form of entry can also be used on the computer on which the UUCP system is installed.

You can test the UUCP system by mailing a short letter to yourself by means of another site. For example, if you are on the site *chicago*, and there is another **micnet** site named *seattle* in the system, then the following command sends mail to the *seattle* site, then back to your *chicago* site, and finally to the user *johnd* in your **micnet** network:

```
mail seattle!chicago!johnd
```

Note that a UUCP system usually performs its communication tasks according to a fixed schedule and may not return mail immediately.

11.6 Administrating Your UUCP System

This section discusses the various shell scripts that are used to supervise and maintain UUCP. Consult the section on “Administration and Maintenance Commands” for details on all commands available to the system administrator. Included is an extended description of the `/usr/spool/uucp` work directory and a special subsection on troubleshooting.

11.6.1 Scheduling Communications with Other Systems

Scheduled UUCP communications are the result of the complex interaction of two shell scripts, **uudemon.hour** and **uudemon.poll**, and the programs

uusched and **cron**. The following list isolates each of the relationships between these programs and explains how they work together to schedule jobs:

- The **uusched** program schedules the queued work in the spool directory, choosing at random the order in which remote computers will be called, before starting the **uucico** daemon.
- The shell script **uudemon.hour** executes **uusched** twice per hour to check for work files.
- The **uudemon.hour** script is in turn started by **cron**, which checks the file every two minutes to see if it is time.
- In the case of polled sites (described below), the script **uudemon.poll** is called by **cron** and sets up a work file started by **uudemon.hour**.

While in multi-user mode, **cron** scans files in */usr/spool/cron/crontabs* once each minute for entries to execute at this time. As the system administrator, you should familiarize yourself with **cron(C)** and the two **uudemon** shell scripts discussed here and two others, **uudemon.admin** and **uudemon.clean**, that are discussed later.

An example crontabs file, *crontab.eg*, is provided to activate these daemons. The system administrator should copy these from */usr/lib/uucp* to */usr/spool/cron/crontabs/root*.

How Often the UUCP Directory is Checked for Work

The **uudemon.hour** shell script does the following:

- Calls the **uusched** program to search the spool directories for work files (C.) that have not been processed and schedules these files for transfer to a remote machine.
- Calls the **uuxqt** daemon to search the spool directories for execute files (X.) that have been transferred to your computer and were not processed at the time they were transferred.

The following is the default, root *crontab* entry for **uudemon.hour**:

```
39, 9 * * * * /usr/lib/uucp/uudemon.hour > /dev/null
```

This script runs twice per hour (at 39 and 9 minutes past). Thus, you may want to execute it more often if you expect higher failure rates.

Polling Remote Computers



As mentioned in the discussion of passive systems and the *Systems* file, it is possible to configure a site that originates no calls and the traffic flow is always one-way. Such sites require *polling*, where other sites periodically dial in (whether or not they have actual transactions logged) to check for jobs.

The *Poll* file (*/usr/lib/uucp/Poll*) contains information for polling remote computers. Each entry in the **Poll** file contains the name of a remote computer to call, followed by a tab character (a space won't work), and finally the hours the computer should be called. The hours must be integers in the range 0-23.

Poll file entries have the following format:

```
sysname<TAB>hour ...
```

For example, the following entry will provide polling of computer *gorgon* every four hours:

```
gorgon 0 4 8 12 16 20
```

The **uudemon.poll** script controls polling but does not actually perform the poll. It merely sets up a polling file (**C.sysnxxxx**) in the */usr/spool/uucp/nodename* directory, where *nodename* is replaced by the name of the machine. This file will in turn be acted upon by the scheduler (started by **uudemon.hour**). The **uudemon.poll** script is scheduled to run twice an hour just before **uudemon.hour** so that the work files will be 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"
```

11.6.2 Automated Maintenance

The UUCP distribution includes predefined entries in the example file */usr/lib/uucp/crontab.eg*, which can be copied to */usr/spool/cron/crontabs/root* to enable these daemons. These entries will automatically perform some administrative tasks for you. The shell scripts are located in */usr/lib/uucp*.

uudemon.admin

The **uudemon.admin** shell script, as delivered, does the following:

- Runs the **uustat** command with **-p** and **-q** options. The **-q** reports on the status of work files (C.), data files (D.), and execute files (X.) that are queued. The **-p** prints process information for networking processes listed in the lock files (`/usr/spool/locks`).
- Sends resulting status information to the UUCP administrative login (**uucp**) via mail.

The default crontab entry for **uudemon.admin** is:

```
48 10,14 * * 1-5 /bin/su uucp -c \  
    "/usr/lib/uucp/uudemon.admin" > /dev/null
```

uudemon.clean

The **uudemon.clean** shell script, as delivered, does the following:

- Takes log files for individual machines from the `/usr/spool/Log` directory, merges them, and places them in the `/usr/spool/Old` directory with other old log information. If log files get large, the `ulimit` may need to be increased.
- Removes work files (C.) 7 days or older, data files (D.) 7 days old or older, and execute files (X.) 2 days old or older from the spool files.
- Mails a summary of the status information gathered during the current day to the UUCP administrative login (**uucp**).

The default crontab entry for **uudemon.clean** is:

```
45 23 * * * ulimit 5000; /bin/su uucp -c \  
    "/usr/lib/uucp/uudemon.clean" > /dev/null
```

Manual Maintenance



Some files may grow indirectly from UUCP and other networking activities. There are two files you should check and delete if they become too large:

<code>/usr/adm/sulog</code>	This file keeps a history of all super user commands. Since uudemon entries in the <code>/usr/cron/root</code> file use the su command, the sulog will grow over time. You should delete this file if it becomes too large.
<code>/usr/lib/cron/log</code>	This file is a log of cron activities. While it grows with use, it is automatically truncated when the system comes up multiuser.

11.6.3 Generating Log Reports on UUCP Usage: **uulog**

The **uulog** program displays log information on UUCP usage according to remote machine. All usage of the programs UUCP, **uuto**, and **uux** are logged in special log files, one per machine.

uulog Options

The **uulog** command has the following options:

Option	Description
<code>-fsystem</code>	Displays the last entry or entries of the <i>system</i> file transfer log.
<code>-ssystem</code>	Displays the <i>system</i> file transfer information.
<code>-x</code>	Displays the uuxqt log file for the given system.
<code>-number</code>	Specifies the <i>number</i> of lines displayed by the <code>-f</code> option.

For example, to print the last 10 lines of *chicago*'s file-transfer log, you would enter

```
uulog -fchicago -10
```

Special uulog Files

During execution of the **uulog** program, the files from the following directories are examined:

Directory	Description
<i>/usr/spool/uucpl.Log/uucicol*</i>	Directory used for queries by the UUCP program.
<i>/usr/spool/uucpl.Log/uuxqt*</i>	Directory used for queries by the uuxqt program.

11.6.4 The UUCP Spool Directory

The following is a comprehensive discussion of all files and subdirectories of the UUCP spool directory. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

TM. (temporary data file)

These data files are created by UUCP processes under the spool directory (i.e., */usr/spool/uucpl/system*) when a file is received from another computer. The *system* directory has the same name as the remote computer that is sending the file. The names of the temporary data files have the format:

TM.*pid.ddd*

where *pid* is a process-ID and *ddd* is a sequential three digit number starting at 0.

When the entire file is received, the **TM.***pid.ddd* file is moved to the path name specified in the *C.synxxx* file (discussed below) that caused the transmission. If processing is abnormally terminated, the **TM.***pid.ddd* file may remain in the *system* directory. These files should be automatically removed by **uucleanup**.



LCK. (lock file)

Lock files are created in the `/usr/spool/uucp` directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The names of lock files have the format:

LCK..*str*

where *str* is either a device or computer name. These files may remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files will be ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock. The lock file is always named using the "a" (non-modem control) suffix to avoid possible conflicts if the same line is specified both modem-control and non-modem-control. For example, the lock on `/dev/tty1A` is named `LCK..tty1a`.

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 computer. The names of work files have the format:

C.*sysnxxx*

where *sys* is the name of the remote computer, *n* is the ASCII character representing the grade (priority) of the work, and *xxx* is the four digit job sequence number assigned by UUCP. Work files contain the following information:

- Full pathname of the file to be sent or requested
- Full 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 (**D.0**) is 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 it is specified in the command line to copy the source file to the spool directory. The names of data files have the following format:

D.*systemxxxxyyy*

where *system* is the first five characters in the name of the remote computer, *xxxx* is a four-digit job sequence number assigned by **uucp**. The four digit job sequence number may be followed by a sub-sequence number, *yyy* that is used when there are several **D.** files created for a work (**C.**) file.

X. (execute file)

Execute files are created in the spool directory prior to remote command executions. The names of execute files have the following format:

X.*sysnxxxx*

where *sys* is the name of the remote computer, *n* is the character representing the grade (priority) of the work, and *xxxx* is a four digit sequence number assigned by UUCP. Execute files contain the following information:

- Requester's login and computer name.
- Name of file(s) required for execution.
- Input to be used as the standard input to the command string.
- Computer and file name to receive standard output from the command execution.
- Command string.
- Option lines for return status requests.

11.7 Troubleshooting

The procedures that follow describe how to solve common UUCP problems.

11.7.1 Check for Faulty ACU/Modem



There are two ways you can check if the automatic call units or modems are not working correctly:

- Run **uustat -q**. This command yields counts and reasons for contact failure.
- Run **cu -x9 -l*line dir***. This permits you to use a specific line and print debugging information during the attempt. Note that this command is only permitted for those who have write access to the *Devices* file, in order to protect the modem from interference from unqualified users.

11.7.2 Check the Systems File

If you are having trouble contacting a particular machine, ensure that the information in your *Systems* file is current. Some things that could be out of date are:

- Phone number
- Login
- Password

11.7.3 Debug Transmissions

If you are unable to contact a particular machine, you can check out communications to that machine using **uutry** and **uucp**. Do the following:

1. Make contact using this command line:

```
/usr/lib/uucp/uutry -r machine
```

where *machine* is the node name of the problem machine. This command does the following:

- Starts the transfer daemon (**uucico**) with debugging. You will get more debugging information if you are **root**.

- Directs the debugging output to `/tmp/machine`.
- Prints the debugging output to your terminal (**tail -f**). Press the DEL key to end output.



You can copy the output from `/tmp/machine` if you wish to save it.

2. if **uutry** fails to isolate the problem, attempt to queue a job with the following command:

```
uucp -r file machine!/dir/file
```

where *file* is the file you want to transfer, and *machine* is the machine you want to copy to, and *dir/file* is the destination location on the other machine. The **-r** option will queue a job without starting a transfer.

3. Next, use **uutry** again. If you still cannot solve the problem, you may need to call support personnel. Save the debugging output; it will help diagnose the problem.

11.7.4 Check Basic Information

There are several commands you can use to check for basic communication information:

uname	Use this command to list the machines you are set up to contact.
uulog	Use this command to display the contents of the log directories for particular hosts.
uucheck -v	Run this command to check for the presence of files and directories needed by uucp . This command also checks the <i>Permissions</i> file and outputs information on the permissions you have set set up.

11.8 Keeping Traffic and Congestion Under Control

The UUCP filesystem can be choked by traffic if a connection goes down, but unless your site is running a full USENET feed or your system connects with a number of systems, UUCP should prove self-sustaining. If UUCP is used more frequently on your system, this section discusses how to ensure that the system does not become stopped, congested, or affect the general performance of your system.

11.8.1 Crowded Directories and Lack of Space



The **uudemon.clean** script is the best way to prevent the UUCP spool directory from growing too large. To see how much disk storage is currently used by UUCP, use the **du(C)** command:

```
du /usr/spool/uucp /usr/spool/uucppublic
```

The current amount of disk space used in each directory is displayed in 512-byte blocks. Divide this number by two for the size in 1K bytes.

The **uudemon.admin** and **uudemon.clean** scripts will send great deal of mail to the **uucp** account. You should check and clear the mail file periodically.

11.8.2 Running out of Processes

On systems with a large amount of traffic, you may get error messages indicating that there are too many processes. If you use the **ps(C)** command, you may notice a number of **uucico** or **uuxqt** processes running. You can establish a new limit on the number of these processes by editing the files *Maxuuscheds* and *Maxuuxqts* in */usr/lib/uucp*.

11.8.3 Evaluating Apparent Stoppages

If users complain that UUCP mail is not getting through and the spool directory is filled with old jobs, it is time to check for the source of the stoppage. UUCP provides an extensive set of error messages and log files that should allow you to trace the cause and remedy the situation.

- Use the **uulog(ADM)** command to study traffic on a per-system basis. Refer to “Generating Log Reports on UUCP Usage: uulog” for details. Error messages in the *Admin/errors* are called ASSERT errors. These usually involve filesystem problems.
- Find out the status of currently queued jobs using the **uustat -q** command. This command will also indicate the number of failed connection attempts.

Error messages are explained in “UUCP Error Messages” in this chapter. Each message is documented with a suggested remedy.

11.9 UUCP Error Messages

This section lists the error messages associated with UUCP. There are two types of error messages. ASSERT errors are recorded in the */usr/spool/uucpl.Admin/errors* file. STATUS errors are recorded in individual machine files found in the */usr/spool/uucpl.Status* directory.



11.9.1 ASSERT Error Messages

When a process is aborted, ASSERT error messages are recorded in */usr/spool/uucpl.Admin/errors*. These messages include the file name, sccsid, line number, and the text listed below. In most cases, these errors are the result of filesystem problems. The “erno” (when present) should be used to investigate the problem. If “erno” is present in a message, it is shown as () in the following list.

Error Message	Description/Action
CAN'T OPEN	An open() or fopen() failed. Check for the presence of the file and permissions.
CAN'T WRITE	A write(), fwrite(), fprintf(), etc. failed. Check for the presence of the file and permissions.
CAN'T READ	A read(), fgets(), etc. failed. Check for the presence of the file and permissions.
CAN'T CREATE	A create() call failed. Check permissions.
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. Check for the presence of the file and permissions.
CAN'T CHMOD	A chmod() call failed. Check for the presence of the file and permissions.
CAN'T LINK	A link() call failed. Check for the presence of the file and permissions.
CAN'T CHDIR	A chdir() call failed. Check for the presence of the file and permissions.



Error Message	Description/Action
CAN' T UNLINK	A 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 <i>/usr/spool/uucpl/Corrupt</i> 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 <i>/etc/passwd</i> file. The filesystem is in trouble, or the <i>/etc/passwd</i> 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 <i>Devices</i> 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 <i>gename.c</i> overflowed. A big/strange request was attempted.
TOO MANY SAVED C FILES	Same as previous.
RETURN FROM <i>fixline ioctl1</i>	An <i>ioctl1</i> , which should never fail, failed. There is a system driver problem.

Error Message	Description/Action
BAD SPEED	A bad line speed appears in the <i>Devices/Systems</i> files (Class field).
PERMISSIONS file: BAD OPTION	There is a bad line or option in the <i>Permissions</i> file.
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.
SYSTAT OPEN FAIL	There is a problem with the modes of <i>/usr/lib/uucp/.Status</i> , 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 suppose 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.



11.9.2 UUCP STATUS Error Messages

Status error messages are messages that are stored in the */usr/spool/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.

XENIX System Administrator's Guide

Error Message

Description/Action

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 <i>Devices</i> file for the particular system. Check the <i>Systems</i> 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 <i>Systems</i> 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 <i>dialer-token</i> 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 <i>Permissions</i> 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 <i>/usr/spool/uucp/.Admin/errors</i> file for the error message and refer to the section <i>ASSERT Error Messages</i> .
SYSTEM NOT IN <i>Systems</i>	The system is not in the <i>Systems</i> file.

Building a Remote Network with UUCP

Error Message	Description/Action
CAN'T ACCESS DEVICE	The device tried does not exist or the modes are wrong. Check the appropriate entries in the <i>Systems</i> and <i>Devices</i> 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 UUCP, the process that was talking to your machine may have failed leaving the LCK file. If they have the new version of UUCP, 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 <i>Systems</i> 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 UUCP.
STARTUP FAILED	Login succeeded, but initial handshake failed. Check communication parameters: data word size, parity, stop bits, etc.
CALLER SCRIPT FAILED	This is usually the same as <code>DIAL FAILED</code> . However, if it occurs often, suspect the caller script in the <i>dialers</i> file. Use <code>uutry</code> to check.





Chapter 12

Building a Local

Network With Micnet

- 12.1 Introduction 12-1
- 12.2 Planning a Network 12-1
 - 12.2.1 Choosing Machine Names 12-2
 - 12.2.2 Choosing a Network Topology 12-2
 - 12.2.3 Drawing a Network Topology Map 12-3
 - 12.2.4 Network Connection Strategy 12-3
 - 12.2.5 Assigning Lines and Speeds 12-5
 - 12.2.6 Choosing Aliases 12-7
- 12.3 Building a Network 12-8
 - 12.3.1 Creating the Micnet Files 12-8
 - 12.3.2 Saving the Micnet Files 12-13
 - 12.3.3 Restoring Micnet Files 12-14
- 12.4 Starting the Network 12-16
- 12.5 Testing a Micnet Network 12-17
 - 12.5.1 Checking the Network Connections 12-17
 - 12.5.2 Using the LOG File to Locate a Problem 12-18
 - 12.5.3 Stopping the Network 12-19
 - 12.5.4 Modifying the Micnet Network 12-20
- 12.6 Using a UUCP System 12-20



12.1 Introduction

A Micnet network allows communications between two or more independent XENIX systems. The network consists of computers connected by serial communication lines (that is, RS-232 ports connected by cable). Each computer in the network runs as an independent system, but allows users to communicate with the other computers in the network through the **mail**, **rcp**, and **remote** commands. These commands pass information such as mail, files, and even other commands, from one computer to another.

It is the system manager's task to build and maintain a Micnet network. The system manager decides how the computers are to be connected, makes the actual physical connections, then uses the **netutil** program to define and start the network.

This chapter explains how to plan a network and then build it with the **netutil** program. In particular, it describes:

- How to choose machine names and aliases
- How to draw the network topology map
- How to assign serial lines
- How to create the Micnet files
- How to distribute the Micnet files
- How to test the Micnet network

12.2 Planning a Network

To build a Micnet network, the **netutil** program requires that you provide the names of the computers that are to be in the network, a description of how the computers are to be connected, a list of the serial lines to be used, the names of the users who use the network, and what aliases (if any) they are known by.

To keep the task as simple as possible, you should take some time to plan the network and make lists of the information you are required to supply. To help you make these lists, the following sections suggest ways to plan a network.

12.2.1 Choosing Machine Names

A Micnet network requires that each computer in the network have a unique "machine name." A machine name helps distinguish each computer from other computers in the network. It is best to choose machine names as the first step in planning the network. This prevents confusion later on, when you build the network with the **netutil** program.

12

A machine name should suggest the location of the computer or the people who use it. You can also use any name you wish. The name must be unique and consist of letters and digits. The Micnet programs use only the first eight characters of each name, so be sure those characters are unique.

The **netutil** program saves the machine name of a computer in a *letc/systemid* file. One file is created for each computer. After you have built and installed the network, you can find out the machine name of the computer you are using by displaying the contents of this file.

12.2.2 Choosing a Network Topology

The network topology is a description of how the computers in the network are connected. In any Micnet network, there are two general topologies from which all topologies can be constructed. These are "star" and "linear."

In a star topology, all computers are directly connected to a central computer. All communications pass through the central computer to the desired destination.

In a linear topology, the computers form a chain, with each computer directly connected to no more than two others. All communications pass down the chain to the desired destination.

A network may be strictly star, strictly linear, or a combination of star and linear topologies. The only restriction is that no network may form a ring. For example, you cannot close up a linear network by connecting the two computers at each end.

The kind of topology you choose depends on the number of computers you have to connect, how quickly you want communications to proceed, and how you want to distribute the task of passing along communications. A star topology provides fast communication between computers, but requires both a large portion of the central computer's total operation time and a large number of serial lines on the central computer. A linear topology distributes the communication burden evenly, requiring only two

serial lines per computer, but is slow if the chain is very long (communication between computers can take several minutes). Often a combination of star and linear topologies makes the best network. In any case, make the choice you think is best. If you discover you have made a wrong choice, you may change the network at any time.

12.2.3 Drawing a Network Topology Map

A network topology map is a sketch of the connections between computers in the network. You use the map to plan the number and location of the serial lines used to make the network.

You can make the map while you work out the topology. Simply arrange the machine names of each computer in the network on paper, then mark each pair of computers you wish to connect with serial lines. For example, the topology map for a linear network of three computers might look like this:

```
a ----- b ----- c
```

As you draw, make sure that there is no more than one connection between any two computers in the network. Furthermore, make sure that no rings are formed (a ring is a series of connections that form a closed circle). Multiple connections and rings are not permitted.

12.2.4 Network Connection Strategy

Once you have made the topology map, you can decide which serial ports to use to connect the computers. Since every connection between computers in the network requires exactly two ports (one on each computer) and one serial wire, you need to be very careful about assigning the lines.

Make a list of the serial ports (also called TTY ports) available for use on each computer in the network. You can display a list of the serial ports on a computer by examining the file */etc/ttys*. A port is available if it is not connected to any device such as a terminal or modem.

For example, in our above topology, computer *b* has two network connections, one to computer *a* and one to computer *c*. You would need to allocate two tty ports on computer *b*, one for computer *a* and one for computer *c*.

There is a restriction on tty names that can be used with micnet. No tty port number may be duplicated across the network. Most computers have

12 similar tty numbering schemes (tty1a, tty2a, tty3a, etc.) This means that on every computer in your network, the serial ports have the same names. For example, if you connect tty1b on computer *a* to tty5d on computer *b*, you cannot then assign tty1b on computer *c* to any other machine on your network. The reason for this restriction is that micnet programs do not distinguish between different computers on the network and you must specify the ports used on each computer for communication. Therefore, if the program finds two ttys of the same name in the topology file, the network cannot operate correctly. We recommend a strategy to allow you to easily manipulate micnet connections.

We suggest that you choose a range of numbers that your system does not use and make those nonexistent tty ports into links for micnet. We recommend using tty numbers beginning with 40 or 50. There is no upper limit and you can have as many of these "virtual" ttys as you need in the topology file. For example, since no two computers can use the same tty name in the topology, state in the topology file that computer *a* uses *tty50* to connect to *tty51* on computer *b*. Since these ttys do not exist on any of your systems, you can be certain that the names are not duplicated across the network.

To make these new virtual tty names correspond to actual ttys on your computer, you use the **ln(C)** command to link the filenames you have given in your topology to real ttys on your system. When you use **ln(C)** to link the two filenames, you are telling XENIX that the actual tty is now also known by the new name you give it. When you list the contents of the */dev* directory, you see both names, but both names access the same physical device.

For an example use of **ln(C)**, the following command links *tty50* on computer *a* to whatever physical tty you actually connect the wire. If you plug the wire into *tty1a* on computer *a* and the other end of the wire goes into *tty5d* on computer *b*, you give the following command on computer *a*:

```
ln /dev/tty1a /dev/tty50
```

Then, when micnet sends information to */dev/tty50*, the information actually goes out through *tty1a*. Similarly, on computer *b* link */dev/tty5d*, where you have actually connected the wire, to the virtual tty called *tty51* as follows:

```
ln /dev/tty5d /dev/tty51
```

This allows you to physically connect the wires to whatever ports are available without concern for duplicating connections. For example, if the

only port available on computer *c* was `/dev/tty1a`, you would not have to change your system configuration to use that name in the network topology.

This strategy is also useful when a port fails for some reason. Instead of having to make and distribute a new network topology, you merely change the link from your virtual tty (`tty50`, `51`, `52`, etc.) to some other physical tty (`tty2c` for example) and your network will operate correctly.



12.2.5 Assigning Lines and Speeds

Follow these steps to create and assign your ttys and ports:

- Using the topology map and the strategy outlined above, assign one (and only one) available tty to each network connection for each computer. List both the actual and the virtual ttys you plan to use. For example, if computer *a* has only one available serial line (`tty1a`) and you plan to use the virtual tty name `tty50`, then the entry in the topology map should look like this:

```

a ----- b ----- c
tty1b
(tty50)
```

- Repeat step 1 for all computers in the topology map. Make sure that each connection is assigned a line and a virtual tty and that no two connections on any given computer have the same virtual tty number. When finished, the map should look like this:

```

a           ----- b -----           c
tty1b      tty2a  tty3a      tty1b
(tty50)    (tty51) (tty52)    (tty53)
```

Note

For an example of a star topology, imagine a cartwheel. One computer is the hub of the wheel and the spokes of the wheel are connections to the other network computers. The center machine in a star topology is often called the "hub" machine. The hub machine must have enough available serial ports to allow a connection to every machine on the network.

If a computer does not have enough available serial ports to meet its needs, you can make the lines available by removing the devices already connected to them. If you cannot remove devices, you must redraw your topology map.

- Using the topology map, assign a serial line transmission speed for each computer pair. The speed must be within the normal range for XENIX serial lines (typically 110 to 9600). Transmission speeds are a matter of preference. In general, a higher speed means a smaller amount of time to complete a transmission, but a greater demand on system's input and output capabilities. In some cases, transmission speeds are a matter of hardware capabilities. Some hardware is not capable of transmission speeds greater than 1200 baud. For this reason, 1200 is the recommended speed when first installing Micnet. You may then increase the speed if you find the hardware can support it.
- After the topology map is completely filled in, make a list of all computer pairs, showing their machine names, serial lines, and transmission speeds. You will use this list when installing the network. Here is the topology map showing the default transmission speeds:

computer	computer	computer
a	---1200--- b ---1200---	c
tty1b	tty2a tty3a	tty1b
(tty50)	(tty51) (tty52)	(tty53)

Here is a sample list of computer pairs from the above topology:

```
a (tty50) to b (tty51) at 1200 baud
b (tty52) to c (tty53) at 1200 baud
```

5. Go to each computer on the network and give the commands to link each virtual tty to its physical counterpart on that machine. For example, the commands should have the form:

In /dev/actual-tty /dev/virtual-tty

12.2.6 Choosing Aliases

Once you have decided how to connect the computers in the network, you can choose aliases for users in the network. An alias is a simple name that represents both a location (computer) and a user. Aliases are used by the **mail** command to allow you to refer to specific computers and users in a network without giving the explicit machine and user names. Although not a required part of the network, aliases can make the network easier to use and maintain.

There are three kinds of aliases: standard, machine, and forward. A standard alias is a name for a single user or a group of users. A machine alias is a name for a computer or an entire network (called a site). A forward alias is a temporary alias for a single user or group of users. A forward alias allows users who normally receive network communications at one computer to receive them at another.

When you build a network with the **netutil** program, you are asked to provide standard aliases only. (You can incorporate machine and forward aliases into the network at your leisure.) Each standard alias must have a unique name and a list of the login names of the users it represents. You may choose any name you wish as long as it consists of letters and numbers, begins with a letter, and does not have the same spelling as the login names. The name should suggest the user or group of users it represents. The login names must be the valid login names of users in the network.

To help you prepare the aliases for entry during the **netutil** program, follow these steps:

1. Make a list of the user aliases (that is, the aliases that refer to just one user) and the corresponding login names of each user.
2. Make a separate list of the group aliases (that is, the aliases that refer to two or more users) and the login names or user aliases (from the first list) of the corresponding users. A group alias may have any number of corresponding users.

Note that there are a number of predefined group aliases. The name **all** is the predefined alias for all users in the network. The machine names of the computers in the network are predefined aliases for the users on each computer. Do not use these names when defining your own aliases.

12.3 Building a Network

You build a network with the **netutil** program. The program allows you to define the machines, users, and serial lines that make up the network.

To build a network, you must first create the Micnet files that define the network and then transfer these files to each computer in the network. After each computer receives the files, you may start the network and use it to communicate between computers.

The following sections describe how to build the network.

12.3.1 Creating the Micnet Files

The Micnet files are created with the **install** option of the **netutil** program. The **install** option asks for the names, aliases, and serial lines of each computer in the network. As you supply the information, it automatically creates the files needed for each computer. These files can then be transferred to the other computers in the network with the **save** and **restore** options of **netutil**. This means you can build the entire network from just one computer.

To use the **install** option, follow these steps:

1. Log in as the super user.
2. Enter:

netutil

and press the **RETURN** key. The program displays the network utility menu. The **install** option is the first item in the menu.

3. Enter "1", and press **RETURN**. The program displays the following message:

```
Compiling new network topology
Overwrite existing network files? (yes/no)?
```

12

Enter **y** and press the **RETURN** key to overwrite the files. The existing network files must be overwritten to create the new network. The first time you install the network, these files contain default information that need not be saved. If you install the system a second time or expand the system, it may be wise to save a copy of these files before starting the **install** option. The files can be saved on a floppy or a hard disk with the **save** option described later in this chapter.

Once you have entered **y**, the program displays the following message:

```
Enter the name of each machine
(or press RETURN to continue installation).
Machine name:
```

4. Enter the machine name and press the **RETURN** key. You may enter more than one name on a line by separating each with a comma or a space. After you have entered all the names, press the **RETURN** key to continue to the next step. The program displays the names you entered and asks if you wish to make changes.
5. Enter **y** (for "yes") if you wish to enter all the names again. Otherwise, enter **n** (for "no") or just press the **RETURN** key to move on to the next step. If you enter **n**, the program displays the message:

```
For each machine, enter the names of the machines
to be connected with it
Machine a:
Connect to:
```

- Using the list of machine pairs you created when planning the network, enter the machine names of the computers connected to the given computer. You may enter more than one name on a line by separating each name with a comma (,) or a space. When you have entered the machine names of all computers connected to the given computer, press the **RETURN** key. The program prompts for the names of the computers connected to the next computer.
- Repeat step 5 for all remaining computers. As the program prompts for each new set of connections, it shows a list of the machine names it already knows to be connected with the current computer. You need not enter these names. The program automatically checks for loops. If it finds one, it ignores the machine name that creates the loop and prompts for another.

Finally, when you have given the connections for all computers in the network, the program displays a list of the connections and asks if you wish to make corrections.

- Enter *y*, if you wish to enter the connections again. Otherwise, enter *n*, to move to the next step. If you enter *n*, the program displays the message:

```
For each machine pair, enter the tty name and tty speeds
For the a <==> b machine pair.
  Tty on a:
```

- Using the list of serial line assignments you created when planning the network, enter the serial line name or number (for example, tty03 or 3) for the first computer in the pair and press the **RETURN** key. The program displays the message:

```
Tty on b:
```

10. Enter the serial line name for the second computer in the pair and press the **RETURN** key. The program displays the message:

```
Speed:
```

11. Enter the speed (for example, 1200) and press the **RETURN** key. The program asks for the serial lines and transmission speed of the next pair.
12. Repeat step 8 for all remaining machine pairs. When you have given serial lines and speeds for all pairs, the program displays this information and asks if you wish to make corrections.
13. Enter *y*, if you wish to enter the serial lines and speeds again. Otherwise, enter *n*, to move to the next step.

The program displays the message:

```
Enter the names of users on each machine:
```

```
For machine a:
```

```
Users on a:
```

14. Enter the login name of a user on the given computer, then press the **RETURN** key. You may enter more than one name on a line by separating each name with a comma (,) or a space. When you have entered all names for the given computer, press the **RETURN** key. The program displays the names of the users on the computer and asks if you wish to make corrections.
15. Enter *y*, if you wish to enter the user names again. Otherwise, enter *n*. If you enter *n*, the program prompts you for the names of the users on the next computer.



16. Repeat steps 13 and 14 for all remaining computers. After you have entered the names of users for every computer, the program prompts you to enter any aliases:

```
Do you wish to enter any aliases? (yes/no)?
```

17. Enter *y*, if you wish to enter aliases. Otherwise, enter *n*, to complete the installation. If you enter *y*, the program displays the message:

```
Each alias consists of two parts, the first is the alias name,  
the second is a list of one or more of the following:
```

```
    valid user names  
    previously defined aliases  
    machine names
```

```
Aliases:
```

18. Using the list of aliases you created when planning the network, enter the name of an alias and press the **RETURN** key. The program displays the message:

```
Users/Aliases:
```

19. If the alias is to name a single user, enter the login name of that user and press the **RETURN** key. The program then prompts for another alias.

If, on the other hand, the alias is to name several users, enter the login names of the users. If one or more of the users to be named by the alias are already named by other aliases, enter the aliases instead of the login names. If all the users on one computer are to

be named by the alias, enter the machine name instead of the login names. In any case, make sure that each item entered on the line is separated from the next by a comma (,) or a space. If there are more items than can fit on the line, enter a comma after the last item on that line and press the **RETURN** key. You can then continue on the next line. After all names and aliases have been entered, press the **RETURN** key. The program then prompts you for another alias.

20. Repeat steps 17 and 18 for all remaining user aliases in your list. When you have given all aliases, press the **RETURN** key. The program displays a list of all aliases and their users and asks if you wish to make corrections.
21. Enter *y*, if you wish to enter all aliases again. Otherwise, enter *n*, to complete the installation.

Once you direct **netutil** to complete the installation, it copies the information you have supplied to the network files, displaying the name of each file as it is updated. Once the files are updated, you may use the **save** option to copy the Micnet files to floppy disk.

12.3.2 Saving the Micnet Files

You can save copies of the Micnet files on backup media (floppy disk) or hard disk with the **save** option of the **netutil** program. Saving the files allows you to transfer them to the other computers in the network. Before you can save the files to a floppy you need to format a floppy disk (see the section “Formatting Floppy Disks” in “Using Floppy Disks and Tape Drives”). Saving the files to the hard disk enables you to use **uucp(C)** to transfer the files to other machines.

To save the files, follow these steps:

1. Log in as the super user.
2. Enter:

netutil

Press the **RETURN** key. The program displays the network utility menu.

3. Enter "2", and press the **RETURN** key. The program displays the message:

```
Save to /dev/fdx (yes/no)?
```

where x is a drive number.

4. If you wish to use the specified disk drive, insert a blank, formatted floppy disk into the drive, wait for the drive to accept the disk, then enter "yes", and press the **RETURN** key. If you do not wish to use the drive, enter "no", and press the **RETURN** key. The program displays a prompt asking you for the filename of the disk drive (or file) you wish to use. Insert a blank, formatted disk into your chosen drive, wait for the drive to accept the disk, then enter the filename of the drive. The name of the default backup device (disk drive) is specified in the file */etc/default/micnet*. This device can be changed depending on system configuration.

In either case, the program copies the Micnet files to the floppy disk.

5. Remove the floppy disk from the drive. Using a soft tip marker (do not use a ball point pen), label the disk "Micnet disk".

As soon as all files have been copied, you can transfer them to all computers in the network.

12.3.3 Restoring Micnet Files

The last step in building a Micnet network is to copy the Micnet files from the Micnet disk to all computers in the network. Do this with the **restore** option of the **netutil** program. For each computer in the network, follow these steps:

1. Log in as the super user.
2. Enter:

```
netutil
```

Press the **RETURN** key. The program displays the network utility menu.

3. Enter the number 3, and press the **RETURN** key. The program displays the message:

```
Restore from /dev/fdx (yes/no)?
```



where x is the number of a drive.

4. If you wish to use the specified disk drive, insert the Micnet disk into the drive, wait for the drive to accept the disk, then enter “yes” and press the **RETURN** key. If you do not wish to use the drive, enter “no” and press the **RETURN** key. The program displays a prompt asking you for the filename of the disk drive you wish to use. Insert the Micnet disk into your chosen drive, wait for the drive to accept the disk, then enter the filename of the drive.

In either case, the program copies the network files to the appropriate directories, displaying the name of each file as it is copied. Finally, the program displays the message:

```
Enter the name of this machine:
```

5. Enter the machine name of the computer you are using and press the **RETURN** key. The program copies this name to the new `/etc/systemid` file for the computer. If necessary, it also disables the serial lines to be used on the computer, preparing them for use with the network.

When the files have been copied, you may start the network with the **start** option.

12.4 Starting the Network

Once the Micnet files have been transferred to a computer, you can start the network with the **start** option of the **netutil** program. The **start** option starts the Micnet programs which perform the tasks needed to communicate between the computers in the network.

12

To start the network, follow these steps for each computer in the network:

1. Log in as the super user.
2. Enter:

netutil

Press the **RETURN** key. The system displays the network utility menu.

3. Enter "4", and press the **RETURN** key. The program searches for the */etc/systemid* file. If it finds the file it starts the network. If it does not, it prompts you to enter the machine name of the computer and then creates the file. The program also asks if you wish to log errors and transmissions. In general, these are not required except when checking or testing the network. When starting the network for the first time, enter *n* in response to each question and press the **RETURN** key.

Once the network has started, you may move to the next computer and start the network there.

Note that, for convenience, you can let each computer start the network automatically whenever the system itself is started. Simply include the command:

netutil start

in the system initialization file, */etc/rc*, of each computer. To add this command, use a text editor as described in the section "Changing the */etc/rc* File" in the "Preparing XENIX for Users" chapter. You can add the **-x** or **-e** options to this command line if you wish to log transmissions or errors. Even if you do not use these options, Micnet copies a log in and log out message to the system *LOG* file each time you start and stop the network. This means you need to periodically clear the file. See the section "Clearing Log Files" in the "Using Filesystems" chapter.

12.5 Testing a Micnet Network

After you have started a network for the first time, you should test the network to see that it is properly installed. In particular, you must determine whether or not each computer is connected to the network.

To test the network, you need to know how to use the **mail(C)** command (see the “Mail” chapter in the *XENIX User's Guide*). The following sections explain how to test the network and how to correct the network if problems are discovered.

12

12.5.1 Checking the Network Connections

You can make sure that all computers are connected to the network by mailing a short message to **all** (the alias for all users in the network) with the **mail** command. Follow these steps:

1. Choose a computer.
2. Log in as the super user.
3. Use the **mail** command (see the *XENIX User's Guide*) and the **all** alias to mail the message:

Micnet test

to all users in the network.

4. Check the mailboxes of each user in the network to see if the message was received. To check the mailboxes, log in as the super user at each computer and use the **cat** command to display the contents of each user's mailbox.

The name of each user's mailbox has the form:

/usr/spool/mail/login-name

where *login-name* is the user's login name.

The network is properly installed when all users have received the message. If the users at one or more computers fail to receive the message, the computers are not properly connected to the network. To fix the problem, you need to locate the computer which has failed to make a connection. The next section explains how to do this.

12.5.2 Using the LOG File to Locate a Problem

You can locate a problem with connections by examining the *LOG* files on each computer in the network. The *LOG* files contain records of the interaction between each pair of computers. There are two *LOG* files for each pair of computers (one file on each computer). The *LOG* files on any given computer are kept in subdirectories of the */usr/spool/micnet* directory. Each subdirectory has as its name the *machine-name* of the other computer in the pair. You can examine the contents of a *LOG* file by entering:

```
cat /usr/spool/micnet/remote/machine-name/LOG
```

and pressing the **RETURN** key. The *machine-name* must be the name of a computer that is paired with the computer you are using.

Each *LOG* file should contain a “startup message” which lists the name of each computer in the pair, and the serial line through which the pair is connected. It also shows the date and time at which the network was started. The message should look like:

```
daemon.mn: running as MASTER
Local system: a
Remote system: b, /dev/tty52
Tue Sep 24 22:30:35 1985
```

A startup message is added to the file each time the network starts successfully. If the message is not present, one or more of the the network files and directories cannot be found. Make sure that you have used the **restore** option to transfer all the network files to the computer. Also, make sure that the */etc/systemid* file contains the correct machine name for the given computer.

Each *LOG* file contains a “handshake” message if the connection between the computer pair has been established. The message:

```
first handshake complete
```

is added to the file on a successful connection. If the message is not present, make sure that the network has been started on the other computer in the pair. The network must be started on both computers before any connection can be made. If the network is started on both computers but the handshake message does not appear, then the serial line may be damaged or improperly connected. Check the serial line to make sure that the cable is firmly seated and attached to the correct RS-232

connectors on both computers. If necessary, replace the cable with one known to work.

If both the startup and handshake messages appear in the *LOG* file but the network is still not working, then there is a problem in transmission. You can create a record of the transmissions and errors encountered while transmitting by restarting the network and requesting Micnet to log all transmissions and errors. Just enter *y* (for “yes”) when the **start** option asks if you wish to log errors or transmissions.

Error entries contain the error messages generated during transmission. Each message lists the cause of the error and the subroutine which detected the error. For example, the message:

```
rsync: bad Probe resp: 68
```

shows that the *rsync* subroutine received a bad response (character 68 hexadecimal) from the other computer. You may use this information to track down the cause of the problem. One common problem is stray information being passed down the serial line by electronic noise. Make sure that the serial line's cable is properly protected against noise (for example, that the cable does not lie near any electric motor, generator, or other source of electromagnetic radiation). Also make sure the cable is in good condition.

Transmission entries contain a record of normal transmissions between computers. Each entry lists the direction, byte count, elapsed time, and time of day of the transmission. For example, the entry:

```
rx: 29349b 2:22 @16:22
```

shows that 29349 bytes were received (*rx*) at 16:22. The elapsed time for the transmission was 2 minutes and 22 second. You can use the records to see if messages are actually being transmitted.

12.5.3 Stopping the Network

You can stop the network with the **stop** option of the **netutil** program. This option stops the Micnet programs, stopping communication between computers in the network.

To stop the network, follow these steps on each computer in the network:

1. Log in as the super user.
2. Enter:

netutil

Press the **RETURN** key. The program displays the network utility menu.

3. Enter "5", and press the **RETURN** key. The program stops the network programs running on the computer.

12.5.4 Modifying the Micnet Network

You can modify a Micnet network at any time by changing one or more of the Micnet files. You can reinstall the network with the **netutil** program. For very small changes (for example, correcting the spelling of an alias), you can modify the Micnet files directly with a text editor. The files and their contents are described in detail in the (M) section of the *XENIX User's Reference*.

Before making any changes to a file, a copy should be made. You can make a copy with the **cp** command. You can replace an old file with the updated file using the **mv** command. Once one or more files have been changed on one computer, the files must be transferred to the other systems in the network using the **save** and **restore** options. These options can only be used after you have stopped the network.

Note that changes to the *aliases* file are not incorporated into the system until the **aliashash** program is executed. This program produces the *aliases.hash* file needed by the network to resolve aliases. See **aliashash(ADM)** in this guide for a description of this command.

12.6 Using a UUCP System

You can send and receive mail from other Micnet sites by installing a UUCP system on one computer in your site. A UUCP system is a set of XENIX programs that provide communication between computers using ordinary telephone lines.

To use a UUCP system with your Micnet network, follow these steps:

1. Install a UUCP system on one computer in the Micnet site. Installation of a uucp system requires a modem and the UUCP software

provided with the XENIX *Operating System*. See the “Building a Remote Network with UUCP” chapter in this Guide for complete details.

2. Add the entry:

```
uucp:
```

to the *aliases* file of the computer on which the UUCP system is installed.

3. For all other computers in your site, add the entry:

```
uucp:machine-name:
```

to the *aliases* file. The *machine-name* must be the name of the computer on which the UUCP system is installed. One may also use the longer form of entry on the computer on which the uucp system is installed.

You can test the **uucp** system by mailing a short letter to yourself via another site. For example, if you are on the site “chicago”, and there is another Micnet site named “seattle” in the system, then the command:

```
mail seattle!chicago!johnd
```

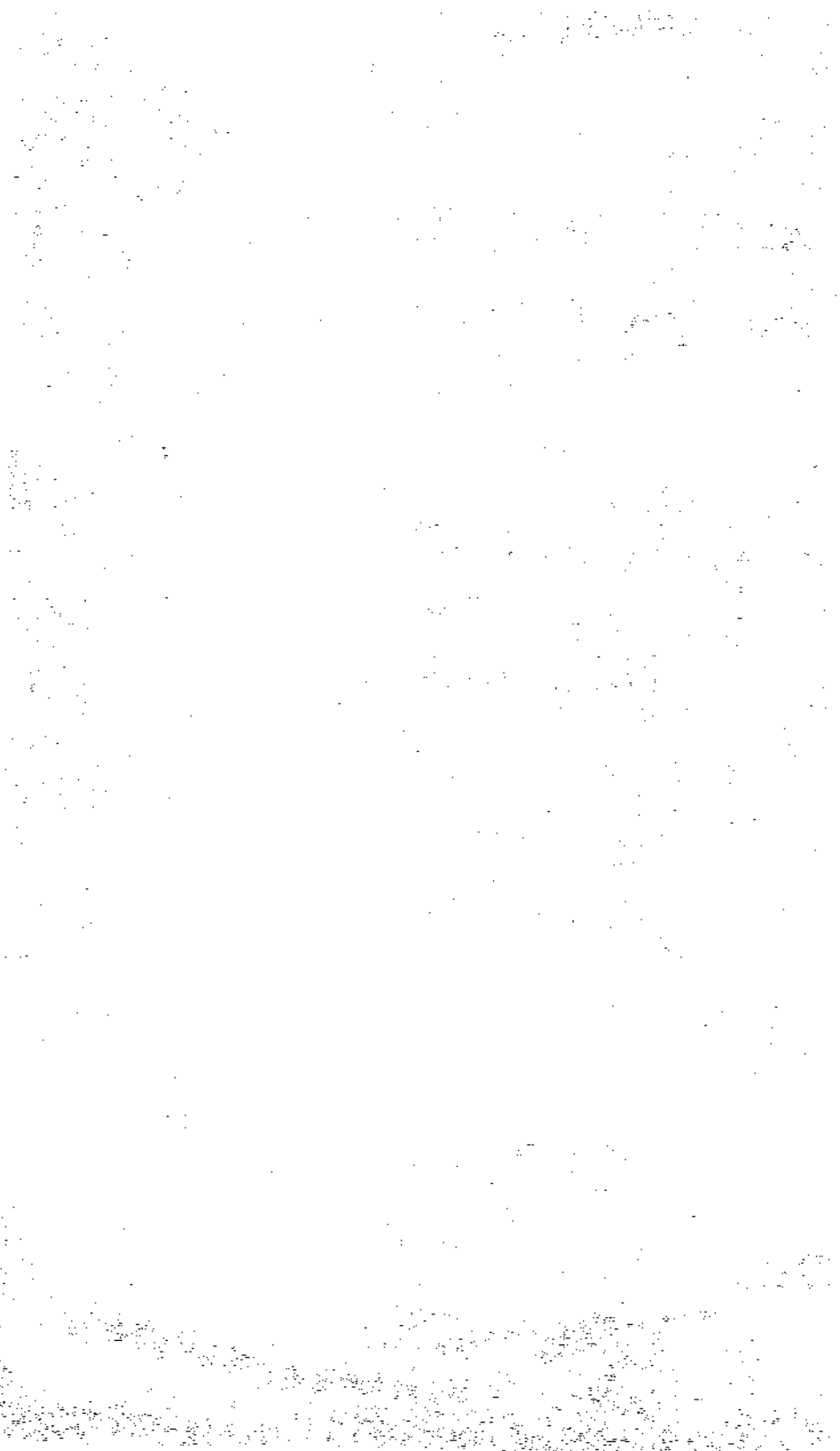
sends mail to the “seattle” site, then back to your “chicago” site, and finally to the user “johnd” in your Micnet network. Note that a UUCP system usually performs its communication tasks according to a fixed schedule, and may not return mail immediately.



Chapter 13

XENIX Directories and Special Device Files

- 13.1 Introduction 13-1
- 13.2 XENIX Directories 13-1
 - 13.2.1 The Root Directory 13-1
 - 13.2.2 The */bin* Directory 13-1
 - 13.2.3 The */dev* Directory 13-2
 - 13.2.4 The */etc* Directory 13-3
 - 13.2.5 The */lib* Directory 13-4
 - 13.2.6 The */mnt* Directory 13-5
 - 13.2.7 The */tmp* Directory 13-5
 - 13.2.8 The */usr* Directory 13-5
- 13.3 Log Files 13-6
- 13.4 Special Device Files 13-7
 - 13.4.1 Special Filenames 13-7
 - 13.4.2 Block Sizes 13-7
 - 13.4.3 Gap and Block Numbers 13-8
 - 13.4.4 Terminal and Network Requirements 13-8



13.1 Introduction

This chapter lists the most frequently used files and directories in the XENIX system. Many of these files and directories are required for proper XENIX operation and must not be removed or modified. The following sections briefly describe each directory.

This chapter also contains information needed to create device nodes relating to filesystems and terminals. For a full description of the special files mentioned here, see section (HW) of this guide.

13.2 XENIX Directories

The following subsections discuss each of the main directories of the XENIX Operating System.



13.2.1 The Root Directory

The root directory (/) contains the following system directories:

/bin	XENIX command directory
/dev	Device special directory
/etc	Additional program and data file directory
/lib	C program library directory
/mnt	Mount directory (reserved for mounted filesystems)
/usr	User service routines (may contain user home directories)
/tmp	Temporary directory (reserved for temporary files created by programs)

All of the above directories are required for system operation.

The root directory also contains a few ordinary files. Of these files, the most notable is the *xenix* file which contains the XENIX kernel image.

13.2.2 The /bin Directory

The */bin* directory contains the most common XENIX commands, that is, the commands likely to be used by anyone on the system. The following is a list of a few of the commands:

XENIX System Administrator's Guide

basename	echo	passwd	su
cp	expr	rm	sync
date	fsck	sh	tar
dump	login	sleep	restor
dumpdir	mv	stty	test

These commands and all others in the */bin* directory are required.

13.2.3 The */dev* Directory

13 The */dev* directory contains special device files which control access to peripheral devices. All files in this directory are required, and must not be removed. There are several subdirectories to the */dev* directory. Each of these subdirectories holds special device files related to a certain type of device. For example, the */dev/dsk* directory contains device files for floppy and hard disks. The files in these directories are linked to the device files that exist in */dev*. You can access the same device through the file in */dev* or the file for the same device in a subdirectory of */dev*.

The following is a list of the files:

<i>/dev/console</i>	System console
<i>/dev/dsk/0s0</i>	Entire disk on drive 0
<i>/dev/dsk/0s1</i>	First disk partition on drive 0
<i>/dev/dsk/0s2</i>	Second disk partition on drive 0
<i>/dev/dsk/1s0</i>	Entire disk on drive 1
<i>/dev/dsk/1s1</i>	First disk partition on drive 1
<i>/dev/dsk/1s2</i>	Second disk partition on drive 0
<i>/dev/dsk/f0d9d</i>	48tpi floppy device
<i>/dev/dsk/f0d9dt</i>	48tpi entire disk floppy device
<i>/dev/dsk/f0q15d</i>	96tpi floppy device
<i>/dev/dsk/f0q15dt</i>	96tpi entire disk floppy device
<i>/dev/rdisk/f0d9d</i>	raw 48tpi floppy device
<i>/dev/rdisk/f0d9dt</i>	raw 48tpi entire disk floppy device
<i>/dev/rdisk/f0q15d</i>	raw 96tpi floppy device
<i>/dev/rdisk/f0q15dt</i>	raw 96tpi entire disk floppy device
<i>/dev/lp</i>	Lineprinter
<i>/dev/kmem</i>	Kernel virtual memory
<i>/dev/mem</i>	Physical memory
<i>/dev/mt/0m</i>	Magnetic tape device
<i>/dev/mt/0mn</i>	Magnetic tape device (no rewind)
<i>/dev/null</i>	Null device (used to redirect unwanted output)

XENIX Directories and Special Device Files

<code>/dev/rmt0</code>	Raw magnetic tape device (linked to <code>/dev/rmt/0m</code>)
<code>/dev/rmt/0m</code>	Raw magnetic tape device
<code>/dev/rmt/0mn</code>	Raw magnetic tape device (no rewind)
<code>/dev/rnn</code>	Unbuffered interface to corresponding device name
<code>/dev/root</code>	Root file structure
<code>/dev/swap</code>	Swap area
<code>/dev/ttynn</code>	Terminals
<code>/dev/tty</code>	The terminal you are using

13.2.4 The `/etc` Directory

The `/etc` directory contains miscellaneous system program and data files. All files are required, but many may be modified.



The following program and data files must not be removed or modified:

<code>/etc/mnttab</code>	Mounted device table
<code>/etc/mount</code>	For mounting a file structure
<code>/etc/mkfs</code>	For creating a file structure
<code>/etc/init</code>	First process after boot

The following data files may be modified, if desired. No files may be removed.

<code>/etc/passwd</code>	Password file
<code>/etc/rc</code>	Bootup shell script
<code>/etc/ttys</code>	Terminal set up
<code>/etc/termcap</code>	Terminal capability map
<code>/etc/motd</code>	Message of the day

The data files in the directory */etc/default* contain default information which is used by system commands (see **default(F)**). The following data files may be modified. No files may be removed.

File	Utility
<i>/etc/default/archive</i>	sysadmin(ADM) default information
<i>/etc/default/backup</i>	backup(C) default information
<i>/etc/default/boot</i>	autoboot(ADM) information
<i>/etc/default/cron</i>	cron(C) default logging information
<i>/etc/default/dumpdir</i>	dumpdir(C) default information
<i>/etc/default/format</i>	format(C) default information
<i>/etc/default/filesys</i>	sysadmin(ADM) default information
<i>/etc/default/login</i>	login(M) default information
<i>/etc/default/lpd</i>	lp(C) default information
<i>/etc/default/mapchan</i>	mapchan(M) default information
<i>/etc/default/micnet</i>	micnet(M) default information
<i>/etc/default/mkuser</i>	mkuser(ADM) default information
<i>/etc/default/msdos</i>	Location of DOS disks (A:, B:,...)
<i>/etc/default/passwd</i>	passwd(C) default information
<i>/etc/default/restor</i>	restore(C) default information
<i>/etc/default/su</i>	su(C) default information (Note that you must create this file yourself.)
<i>/etc/default/tar</i>	tar(C) default information
<i>/etc/default/usemouse</i>	usemouse(C) default information

13.2.5 The */lib* Directory

The */lib* directory contains runtime library files for C and other language programs. The directory is required.

13.2.6 The */mnt* Directory

The */mnt* directory is an empty directory reserved for mounting removable filesystems.

13.2.7 The */tmp* Directory

The */tmp* directory contains temporary files created by XENIX programs. The files are normally present when the corresponding program is running, but may also be left in the directory if the program is prematurely stopped. You may remove any temporary file that does not belong to a running program.



13.2.8 The */usr* Directory

The */usr* directory contains the home directories of all users on the system. It also contains several other directories which provide additional XENIX commands and data files.

The */usr/bin* directory contains more XENIX commands. These commands are less frequently used or considered nonessential to XENIX system operation.

The */usr/include* directory contains header files for compiling C programs.

The */usr/lib* directory contains more libraries and data files used by various XENIX commands.

The */usr/spool* directory contains various directories for storing files to be printed, mailed, or passed through networks.

The */usr/tmp* directory contains more temporary files.

The */usr/adm* directory contains data files associated with system administration and accounting. In particular, the */usr/adm/messages* file contains a record of all error messages sent to the system console. This file is especially useful for locating hardware problems. For example, an unusual number of disk errors on a drive indicates a defective or misaligned drive. Since messages in the file can accumulate rapidly, the file must be deleted periodically.

13.3 Log Files

A variety of directories contain log files that grow in size during the normal course of system operation. Many of these files must be periodically cleared to prevent them from taking up valuable disk space (see the section on "Clearing Log Files" in the "Using Filesystems" chapter). The following table lists the files (by full pathname) and their contents.

Filename	Description
<i>/etc/ddate</i>	Records date of each backup.
<i>/usr/adm/pacct</i>	Records accounting information; grows rapidly when process accounting is on. (See accton (ADM) and accom (ADM).)
<i>/usr/adm/messages</i>	Records error messages generated by the system when started. (See messages (M).)
<i>/etc/wtmp</i>	Records user logins and logouts. (See login (M).)
<i>/usr/adm/sulog</i>	Records each use of the su command; grows only if option is set in the <i>/etc/default/su</i> file. You must create <i>/etc/default/su</i> . (See su (C).)
<i>/usr/lib/cron/cronlog</i>	Records each use of the at (C) and cron (C) commands.
<i>/usr/spool/micnet/remote*/LOG</i>	Records transmissions between machines in a Micnet network. The (*) must be the name of a remote machine connected to the current machine.



13.4 Special Device Files

Many of the filesystem maintenance tasks described in this guide require the use of special filenames, block sizes, and gap and block numbers. The following sections describe each in detail.

13.4.1 Special Filenames

A special filename is the name of the device special block or character I/O file, which corresponds to a peripheral device, such as a hard or floppy disk drive. These names are required in such commands as **mkfs**(ADM), **mount**(ADM), and **df**(C) to specify the device containing the filesystem to be created, mounted, or searched.

The following table lists the special filenames and corresponding devices, for hard and floppy disk drives on a typical computer.

Device Special Filenames - Disks

Filename	Disk Drive
/dev/fd0	Floppy Drive 0
/dev/dsk/f0	Floppy Drive 0
/dev/fd1	Floppy Drive 1
/dev/dsk/f1	Floppy Drive 1
/dev/hd00	Entire hard disk
/dev/dsk/Os0	Entire hard disk
/dev/root	Root filesystem
/dev/usr	User filesystem

13.4.2 Block Sizes

The block size of a disk is the number of blocks of storage space available on the disk, where a block is typically 512 or 1024 bytes of storage. Refer to the **machine**(HW) manual page or use **cmchk**(C) to determine the size of blocks on your system. Many commands require input that defines the number of blocks to be operated on. Other commands report disk space in terms of 512 byte blocks, in particular **df**(C), **du**(C), **ls**(C), **lc**(C), and **find**(C). A 500 byte file on a 1024 byte block filesystem is reported as using 2 blocks by these utilities, as the file uses one system block which is equivalent to two 512 byte blocks. The size of a 10 megabyte hard disk in

1024 byte blocks is 9792. Note that some of the blocks on the disk are reserved for system use and cannot be accessed by user programs. The block size of a typical floppy disk depends on the total storage capacity of the disk, as given by the manufacturer.

13.4.3 Gap and Block Numbers

The gap and block numbers are used by the **mkfs(ADM)**, and **fsck(ADM)**, commands to describe how the blocks are to be arranged on a disk. The following table lists the gap and block numbers for the floppy and hard disks used with a typical computer.

13

<u>Disks</u>	<u>Gap</u>	<u>Block</u>
Floppy Disk, 48ds9	1	9
Floppy Disk, 96ds15	1	15
Floppy Disk, 135ds9	1	9
Floppy Disk, 135ds18	1	18
Hard Disk	1	34

The number of blocks can also be determined by multiplying the number of sectors per track (usually 17) by the number of heads on the hard disk, dividing by 2 (since there are 2 blocks per sector), and rounding off to the nearest integer.

13.4.4 Terminal and Network Requirements

The **enable** and **disable** commands are used to add and remove terminals on a system. The install option of the **netutil** program is used to build a network. The preceding commands and option require the names of the serial lines through which a terminal or network is to be connected. The following table lists the device special filenames of the two serial lines (actually two serial ports either with or without modem control). The character I/O files corresponding to these serial lines can be found in the */dev* directory. Note that the files */dev/console* and */dev/tty01* through */dev/tty12* represent "hardwired" devices and are not available for connection to terminals or hardware. Also, refer to **serial(HW)** for more information on serial lines.

XENIX Directories and Special Device Files

Filename	Line
/dev/tty1a	main serial line (without modem control)
/dev/tty2a	alternate serial line (without modem control)
/dev/tty1A	main serial line (with modem control)
/dev/tty2A	alternate serial line (with modem control)



Chapter 14

Using Terminals and Modems

- 14.1 Introduction 14-1
- 14.2 Using Multiscreen™ 14-1
- 14.3 Adding and Configuring Serial Ports 14-2
- 14.4 Setting Up a Serial Console 14-4
- 14.5 Adding a Terminal 14-5
- 14.6 Setting Terminal Lines 14-9
 - 14.6.1 The gettydefs File 14-9
 - 14.6.2 Changing the gettydefs File 14-11
 - 14.6.3 Checking the Terminal Settings 14-13
- 14.7 Changing Serial Line Operation 14-13
- 14.8 Setting the Terminal Type 14-15
- 14.9 Setting the Terminal Type Automatically 14-16
- 14.10 Removing a Terminal 14-17
- 14.11 Modem Usage under XENIX 14-18
 - 14.11.1 Serial Lines 14-18
 - 14.11.2 Dialing Out From Your Computer 14-19
 - 14.11.3 Installing a Dial-out Modem 14-20
 - 14.11.4 Troubleshooting Your Dial-out Modem 14-22
 - 14.11.5 Dialing Into Your Computer Under XENIX 14-24
 - 14.11.6 Installing a Dial-in Modem 14-24
 - 14.11.7 Troubleshooting Your Dial-in Modem 14-26
 - 14.11.8 Shared Dial-in/Dial-out 14-28
 - 14.11.9 Installing A Shared Dial-in/Dial-out Modem 14-28
 - 14.11.10 Hayes Modem Settings 14-29



14.1 Introduction

One important task of the system administrator is to add peripheral devices such as terminals and modems to the system. Adding these serial devices lets more users access the system and adds to overall system capabilities.

This chapter explains the following tasks:

- Physically connecting serial devices to your computer
- Enabling serial devices for operation
- Maintaining serial devices

Note that physical connections between a device and the system vary according to hardware configuration. For specific information about connecting your serial device, refer to the hardware manuals provided with the device and with your computer.



14.2 Using Multiscreen™

With Multiscreen, you can use your console as several terminals at one time. Pressing a simple key combination switches you from one screen to another, with each screen acting as an independent terminal.

Because each multiscreen is independent, you can log in and run programs on each screen. Because output from your programs is saved in a screen buffer, you see the most recent output for whichever screen you look at. If you stop output to one screen, as when you press the Ctrl-S key combination, only that screen is affected.

The amount of memory in your computer determines the number of multiscreens available on your system. When you boot your system, the number of automatically-enabled multiscreens is displayed. Most machines have between two and six multiscreens enabled, but your machine can have up to twelve if your system has sufficient memory. To increase the number of multiscreens on your system, add to your system's memory; XENIX automatically enables more screens.

Although all of the multiscreens can be open and active at once, you see only one screen at a time. The selected multiscreen is like a terminal that is "connected" to the keyboard. Switching between screens is like moving to another terminal because each multiscreen has its own device file.

The multiple screen feature uses the `/dev/tty[01...12]` device files. These files provide character I/O between your system and your computer screen and keyboard.

To select any active screen, press **alt-Fn**, where **Fn** is one of the function keys on your keyboard. Function keys are generally located across the top or down the far left side of the keyboard. The `tty01` is the “alt-f1” terminal, `tty02` is the “alt-f2” terminal, `tty03` is “alt-f3”, etc. For example, this keystroke switches you to screen 6, corresponding to `/dev/tty06`:

alt-F6

You can also rotate through the screens by pressing the Control and Print Screen key combination, **Ctrl-PrtSc** (using the **Ctrl** key and the **PrtSc** key). Use this combination to access screens for which you do not have function keys: For example, if you have twelve multiscreens enabled, but your computer keyboard has only ten function keys, display screen eleven by pressing alt-F10 to get to screen 10, and then pressing **Ctrl-PrtSc** to rotate to screen 11. To access screen 12, press **Ctrl-PrtSc** again. Pressing **Ctrl-PrtSc** again rotate your back to the first multiscreen, `tty01`.

For more information, refer to **multiscreen(M)** and **screen(HW)**.

14.3 Adding and Configuring Serial Ports

To add a multi-port expansion card, you must first determine whether the card is a “smart” serial card or a standard serial card. If the card is a “smart” card, the manufacturer will have supplied installation software and a driver. These should be all you need to add the card to your XENIX system.

Before installing your card, look through the “Using Bus Cards” chapter in this guide for information you might find helpful and check your XENIX *Release Notes* for information about hardware compatibility with XENIX. Follow the instructions for insertion furnished with your card, referring to your computer hardware manual if necessary.

If your card is a standard serial card, the following instructions explain how to create new device files for additional ports.

1. Boot the system and enter system maintenance mode.

2. When you are in system maintenance mode, enter:

```
/etc/mkdev serial
```

```
Δ sysadmsh users select: System→Add→Card_Serial
```

3. This invokes **serinit**, which begins with the following display:

```
You would like to install a :  
  1. 1 port card  
  2. 4 port card  
  3. 8 port card  
  
Select an option or enter 'q' to quit:
```

Enter the appropriate number and press **RETURN**.

4. The program responds with the following menu (only COM1 and COM2 appear on most systems):

```
The card is configured as:  
  1. COM1  
  2. COM2  
  3. COM3  
  4. COM4  
  
Select an option or enter 'h' for help or 'q' to quit:
```

If you select “h”, you see a table listing ports, card types, I/O addresses, and status addresses.

Enter a number and press **RETURN**. After **serinit** accepts the COM slot, you see a list giving the newly configured ports and their modem control counterparts. For example, **tty2a** and **tty2A** refer to the same serial port, but **tty2A** has modem control, whereas **tty2a** refers to the same port without modem control. You can access the port by only one name at a time, either with or without modem control.

Now that your serial ports are configured, make sure that they are also defined in the XENIX system hardware configuration.

Check your computer hardware manual to determine how your system is configured. If your system is configured using a CMOS database, the ports are defined in the database (see **cmos(HW)**).

If your system is configured with switch settings on the main system board, define the new ports by setting the proper switches (refer to your hardware manuals for the settings).

Note

An error message is displayed if you attempt to access a serial port that has not been installed and defined.

14.4 Setting Up a Serial Console

You can configure a serial device, rather than a display adapter, as your system console. The **boot** program sets the default console at boot time according to the following procedure:

1. The **boot** program looks for the entry **SYSTTY=*x*** (replace *x* with the name of the system console device) in the */etc/default/boot* file.
2. If the **SYSTTY** entry is not found or the */etc/default/boot* file is not readable, **boot** checks your system for a display adapter and designates it as your system console.
3. If no display adapter is found, **boot** looks for **tty1a**, sets the serial port to 9600 baud, 8 data bits, 1 stop bit, and no parity, and uses it as the system console.

To set up a serial console, create the following entry in your */etc/default/boot* file (replace *x* with "0" for a display adapter or with "1" for a COM1 serial port):

```
SYSTTY=x
```

To change the system console device from the command line, enter **systty=*x*** at the boot prompt (replace *x* with "0" for a display adapter or with "1" for a COM1 serial port). This does not create or change a **SYSTTY** entry in the */etc/default/boot* file.

14.5 Adding a Terminal

Before you add a terminal to your system, look in the hardware manual for your terminal for instructions on connecting the terminal to a serial line. Also, refer to the list of standard serial lines in the “XENIX Directories and Special Device Files” chapter to find the name of your serial line. (If you add a serial card, the possible names of the additional device files are listed in **serial(HW)** in this guide.)

To add a terminal to your system, connect it to an RS-232 serial line and enable it with the **enable(C)** command.

XENIX is compatible with many types of terminals. Look in the **terminals(M)** section of the *XENIX User's Reference* for a comprehensive list of terminals supported by XENIX. Support for terminals is provided through the *etc/termcap* file, which contains the definitions and classifications of keystrokes and control sequences that vary from terminal to terminal. For a description of the *etc/termcap* file, see **termcap(M)** in the *XENIX User's Reference*.

The following steps show how to install a terminal with the standard “COM” serial lines, or with serial expansion cards:

1. This step is for serial expansion cards. If you are adding a terminal directly to a COM port, skip to the next step.

If you are using a supported 4 or 8-port expansion board, check to see if your board is recognized at bootup by checking the XENIX bootup message. If the boot process does not accurately report your board, then the switches on your card are not set properly. Check your board's hardware documentation for the proper switch settings and the *XENIX Release Notes* for the correct addresses. This applies to boards that are listed as supported in the *Release Notes*.

Vendor-supplied drivers may not print a recognition message at boot time. If your serial expansion card is a “smart” card with a vendor-supplied driver, you should not need to run **mkdev serial** to install it. For your system to recognize the new card, run the vendor-supplied installation software.

Configure the interrupts for the two standard COM ports - COM1 as interrupt 4 and COM2 as interrupt 3. Most serial cards use one interrupt per board, so two four-port boards can use COM1 and COM2. Be aware of the requirements of other products and hardware to avoid interrupt conflicts. See **serial(HW)** in this guide for more information on COM1 and COM2.

When a supported card is correctly configured for the desired COM port and recognized at bootup, run this command:

```
/etc/mkdev serial
```

△ **sysadmsh** users select: System→Add→Card_Serial

This creates device files for your extra serial ports.

2. Make sure you are logged in as root in multi-user mode.

Plug in your terminal and turn it on. Set it for 9600 baud, 8 data bits, 1 stop bit, no parity, full duplex, and XON/XOFF handshaking. If your terminal does not work in this mode, look for advice on configuring your terminal in the section "Changing the gettydefs File" later in this chapter and in the **stty(C)** page in the *XENIX User's Reference*.

Some terminals connect with a straight cable directly to the computer. Other terminals connect to a modem. Terminals connected to a modem use a "null modem" or "modem connector," - a cable with pins 2 and 3 crossed. Connect the terminal so that Transmit Data on the serial port is connected to Receive Data on the terminal, and Transmit Data on the terminal is connected to Receive Data on the serial port. Signal Ground should be connected to Signal Ground. Other pins probably do not need to be connected. XENIX requires only that pins 2, 3, and 7 are connected.

For more information on your terminal, refer to your terminal manual or a reference on serial communication.

3. If the port is enabled, press the RETURN key a few times to see if a "login:" prompt appears. If so, you are ready to log in. If not, use the console or a working terminal to log in as the super-user (root), and disable the port with this command:

```
disable ttyname
```

In the previous command, *ttyname* is the device special name of the port in question. Make sure you are using a non-modem control device, for example, */dev/tty1a*, not */dev/tty1A*. For more information on serial port names, see **serial(HW)**, in this guide and the section "Adding and Configuring Serial Ports" in this chapter.

4. Check that the entry for this serial port in the */etc/ttys* file looks like the following (*ttyname* is the name of the device file, for example */dev/tty1a*):

```
0mmttyname
```

If the entry does not look like this example, edit the file to correct it. The first digit in each field is a 0 or a 1, depending on whether the device is disabled or enabled. A 0 means the device is disabled; a 1 means the device is enabled.

5. From the console, as *root*, see if you can redirect output to the terminal by entering:

```
date > /dev/ttyname
```

If you do not see the date printed on the terminal and you are not sure of the correct *ttyname*, try other *ttynames* on that serial port. If you still do not see the date printed on the terminal, there try the following:

- Make certain that the terminal is plugged in.
- Check that the cable is configured correctly. If the serial port you are using has a 25-pin connector (DB-25), read through Step 2 in the preceding set of instructions. Are pins 2, 3, and 7 connected correctly? (Note that pins other than 2, 3, and 7 are probably not used.)

If your system or expansion card has a 9-pin connector (DB-9), you must use a 9-pin to 25-pin connector. Look in your hardware manual for information on 9-pin to 25-pin connections.

- Check your terminal setup configuration. See Step 2 in the preceding set of instructions.
- Check the switches on your serial port. If you are using a multi-port card, try other lines on that card.
- Attach the terminal to a standard serial port (COM1 or COM2) to see if the terminal and cable are working correctly. If you are already using a COM port, try switching to another one.

If you have successfully installed another terminal, switch hardware between the working and the nonworking terminal one piece at a time. This may help you isolate a hardware

problem. Note that some faulty hardware may work under DOS but not under XENIX.

6. When you the date prints on your terminal, enable the port with the following command:

enable *ttname*

The **enable** command starts a **getty** process that displays the following login prompt:

```
login:
```

If you do not see the “login:” prompt, enter the following command to verify that **getty** is running on the port and that the software is configured properly:

ps -t *ttname*

Your screen should display a message similar to the one in the following example, with either “login” or “getty” listed in the “COMMAND” column:

```
PID    TTY    TIME   COMMAND
2557   1a     0:06   getty
```

7. If you have typed the **enable** and **disable** commands many times, it is possible that a new **getty** cannot be spawned on that port. If so, shut the system down, reboot, login as “root” in multiuser mode, and try again.

14.6 Setting Terminal Lines

Your XENIX system can automatically adapt to several different terminal baud rates and settings. The same program that displays the login message, **getty(M)**, reads these terminal line values from a table, trying each setting until one is successful, and the user can log in to the system. This table provides several default settings for different kinds of terminal lines.

On your XENIX system, **getty** automatically executes as part of the login process. The table of terminal settings is found in a file called */etc/gettydefs*. You can edit *gettydefs* to add different sets of terminal characteristics or to change the existing ones.

14.6.1 The *gettydefs* File

The file */etc/gettydefs* contains the information that **getty** uses to set up terminal line characteristics such as baud rate. The file is in the form of a table. Each table entry is divided into five fields. These fields include:

label # *initial-flags* # *final-flags* # *login-prompt* # *next-label* [# *login program*]

The fields are:

- | | |
|----------------------|---|
| <i>label</i> | Identifies the <i>gettydefs</i> entry to getty . This could be a number or a letter. <i>label</i> corresponds to the line mode field in <i>/etc/tty</i> s. init passes the line mode to getty as an argument. |
| <i>initial-flags</i> | Sets terminal line characteristics when getty first establishes the connection. getty recognizes the flags listed in termio(M) , in the <i>XENIX User's Reference</i> . Often the only flag in this field is the one setting the baudrate. Foreexample, B300 would set the speed to 300. |
| <i>final-flags</i> | Sets the terminal line characteristics just before getty executes login . These flags describe the operating characteristics for the line. The baud rate (B) is set again. Other common flags include SANE (a composite flag that sets a number of terminal characteristics to reasonable values), TAB3 (expands tabs with spaces), IXANY (enables any character to restart output), and HUPCL (hangs up line on final close). Flags can be entered in any order. |

login-prompt

Contains the login message that greets users. This field is printed exactly as it is entered, including spaces and tabs. An “@” symbol in the login-prompt field is expanded to the first line in the file */etc/systemid* (unless the ‘@’ is preceded by a ‘\’).

Several character sequences are recognized, including:

\n	Line feed	\t	Tab
\r	Carriage return	\f	Formfeed
\v	Vertical tab	\b	Backspace
\nnn	(3 octal digits)		
	The specified ASCII characters		

14

next-label

Identifies the next label in *gettydefs* for **getty** to try if the current one is not successful. **getty** tries the next label if a user presses the **BREAK** key while attempting to log in to the system. Groups of entries, such as dial-up or TTY lines, should form a closed set so that **getty** cycles back to the original entry if none of the entries is successful.

login-program

The name of the program which actually logs users onto XENIX. The default program is */etc/login*. This field is optional and is specific to XENIX.

If preceded by the keyword “AUTO,” **getty** does not prompt for a user name, but instead uses the device name (i.e. *ty03*) as the user name and immediately executes the *login-program*.

Each field is separated by a pound sign (#), and each entry in *gettydefs* is separated by a blank line.

An entry in *gettydefs* might look like the following:

```
4# B1200 # B1200 SANE TAB3 HUPCL #login: #2 #AUTO /etc/login.new
```

Here is a description of each part of this line:

- The number 4 identifies this entry to **getty**.

- The next field sets the baud rate to 1200.
- The third field indicates the baud rate (B1200), SANE (a composite flag for a number of characteristics), and HUPCL (hangs up line on final close).
- *login*: appears as the login prompt. If this setting is not successful, **getty** proceeds to label 2 in *gettydefs*.
- AUTO attempts to log in the user by executing */etc/login.new*.

If the last entry also contains a *filename*, that *login program* is executed. (Note that the *filename* and the corresponding *login program* are user created.) For example, including a file such as */etc/dial_login* for a line connected to a modem can be used. It would set the user ID, acquire a password, validate the user, and then become the user. It could possibly require a password for the system in addition to an account password and even have a special set of login environment variables included in */etc/default/dial_login*.



14.6.2 Changing the gettydefs File

The file */etc/gettydefs* is on your XENIX system and has sets of entries for the dial-up lines and terminal lines. These different sets correspond to line mode settings in */etc/tty*s. The **init** program passes the line mode as an argument to **getty**.

You can edit *gettydefs* to add new terminal settings or to change existing ones. For example, the settings for terminal lines on your XENIX system might look like the following:

```
4 # B2400 HUPCL # B2400 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 5
5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 6
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 4
```

To change the sample *gettydefs* file so that the first baud rate **getty** attempts is 1200, do the following:

1. Enter a text editor to edit the first line of the file *gettydefs*.
2. Change the first and third fields from B2400 to B1200.
3. Save *gettydefs* and exit the editor.

The sample file should look like the next example:

```
4 # B1200 HUPCL # B1200 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 5
5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 6
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 4
```

14

You can also add additional terminal line settings to *gettydefs*. Flags and permissible values for terminal settings are listed in **stty** (C), in the *XENIX User's Reference*.

When you add a new entry, be sure that the groups of entries in *gettydefs* form a closed set, so the *next-label* field of the last entry directs **getty** back to the first entry in the group.

To add an entry for a baud rate of 300 to the preceding sample *gettydefs* file, follow these steps:

1. Enter a text editor to edit the file */etc/gettydefs*.
2. Locate the point where you want to insert the new settings for *gettydefs*. The order of the entries does not matter; **getty** only looks for the label. In this example, the new entry is the last entry in the file.
3. Insert a carriage return after the last line in the file and enter the following on a new line:

```
#B300HUPCL #B300 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: #4
```

4. To incorporate label 7 into the set of labels, change the *next label* field for entry 6 to 7:

```
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY
#r\n@!login:#7
```

getty is now directed from label 6 to 7, and then back to 4.

5. Exit the text editor, saving the revised *gettydefs* file.

The new *gettydefs* looks like the following:

```
4 # B1200 HUPCL # B1200 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 5
5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 6
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 7
7 # B300 HUPCL # B300 CS8 SANE HUPCL TAB3 ECHOE IXANY #r\n@!login: # 4
```

14.6.3 Checking the Terminal Settings

Each time you change the terminal line settings or add new entries to *gettydefs*, you should check to make sure that the new values make sense to **getty**. To do this you use the command **getty** with the check option, **-c**, and the filename.

14

For example, to check *gettydefs*, enter:

```
getty -c /etc/gettydefs
```

If any of the values and settings in *gettydefs* are not permitted, **getty -c** displays them on your terminal screen.

For more information on **getty** and *gettydefs*, see **getty(M)** and **gettydefs(F)**, in the *XENIX User's Reference*.

14.7 Changing Serial Line Operation

Whenever you enable a terminal with the **enable** command, the system automatically sets the operating characteristics of the serial line to a set of default values. Sometimes these values do not match the values used by the terminal and, therefore, must be changed to allow communication between the system and the terminal. You can display and change the operating characteristics of a serial line with the **stty** (for "set tty") command.

XENIX System Administrator's Guide

You can display the current operating characteristics of a serial line by entering this command at the terminal connected to that line:

```
stty
```

If it is impossible to log in at that terminal, you can use another terminal to display the characteristics. Log in as the super-user at another terminal, and enter:

```
stty < ttyname
```

In the previous command, *specialfile* is the name of the device special file corresponding to the serial line (see the “XENIX Directories and Special Device Files” chapter). For example, this command displays the current characteristics of the serial line named */dev/tty1a*:

```
stty < /dev/tty1a
```

The command displays the baud rate, the parity scheme, and other information about the serial line. This information is explained in the **stty(C)** in the *XENIX User's Reference*.

One common change to a serial line is changing the baud rate. This is usually done from a terminal connected to another serial line because changing the rate disrupts communication between the terminal and system. Before you can change the rate, you need to know the current baud rate of the terminal (review the terminal hardware manual to see how to determine the current baud rate). Once you have the baud rate, log in as the super-user at the other terminal, and enter:

```
stty baud-rate < specialfile
```

where *baud-rate* is the current baud rate of the terminal, and *specialfile* is the name of the device special file corresponding to the serial line you wish to change. The baud rate must be in the set 50, 75, 110, 134, 150, 200, 300, 600, 1200, 2400, 4800, and 9600. For example, the command:

```
stty 9600 < /dev/tty1a
```

changes the baud rate of the serial line */dev/tty1a* to 9600. Note that the “less than” symbol (<) is used for both displaying and setting the serial line from another terminal.

Another common change is changing the way the system processes input and output through the serial line. Such changes are usually made from the terminal connected to the serial line. For example, the command:

stty tabs

causes the system to expand tabs with spaces (used with terminals which do not expand tabs on their own), and the command:

stty echoe

causes the system to remove a deleted character from the terminal screen when you back over it with the **BACKSPACE** key.

Note that the **stty** command may also be used to adapt a serial line to an unusual terminal, to another type of serial device which requires parity generation and detection, or special input and output processing.

For a full description of this command, see **stty(C)** in the *XENIX User's Reference*.



14.8 Setting the Terminal Type

The XENIX system requires that the terminal type be clearly defined before any work is done at the terminal. The preferable method for setting your terminal type is to assign the type to the **TERM** variable, a special environment variable that associates the terminal you are using with a list of characteristics given in the */etc/termcap* file. The characteristics tell the system how to interpret your terminal's keys and how to display data on your terminal screen.

If you are using the Bourne shell (**sh**), the **TERM** assignment has the form:

```
TERM=termtype; export TERM
```

If you are using the C-shell (**csh**), the **TERM** assignment has the form:

```
setenv TERM termtype
```

The *termtype* must be one of the names associated with one of the terminals defined in the */etc/termcap* file. The assignment must be entered at the terminal whose type you are setting.

For example, to set the terminal type to "ansi" from Bourne shell, go to the terminal you wish to set, enter at the shell prompt ("\$\$"):

```
TERM=ansi; export TERM
```

and press the RETURN key. From C-shell, enter at the shell prompt ("%"):

```
setenv TERM ansi
```

and press RETURN.

If you are not sure which name you may use for *termtyp*e, you can view the names by displaying the */etc/termcap* file. To display the file, enter:

```
cat /etc/termcap
```

and press the RETURN key. Because the file is large, use **Ctrl-s** to stop the display at every full screen. You may view more of the file by pressing **Ctrl-q** or the SPACEBAR.

You can let the system define the terminal type automatically whenever you log in by including the TERM assignment in your *.profile* file (see "Changing the *.profile* and *.login* Files" in the "Preparing XENIX for Users" chapter).

For an alternate method of setting your terminal type see **tset(C)** in the *XENIX User's Reference*.

If you let the system set the terminal type, be careful when logging in on terminals that are not the same as your normal terminal. The XENIX system has no way of checking whether or not the terminal assignment is correct for the given terminal and assumes that it is the same as your normal terminal. If it is not, you must set the terminal type manually.

14.9 Setting the Terminal Type Automatically

If you want to have the terminal type set automatically at login time, follow this procedure:

1. Login on the terminal in question and determine which *ttyname* you're using by entering:

```
tty
```

2. Login as **root** and edit the file */etc/ttytype* with a text editor. Change the terminal type field for the line associated with the terminal in question to the terminal type you desire to use. Follow the model for

the console. If you want your terminal type to be set to 'wy50' for */dev/tty1a*, edit */etc/ttytype* as follows:

```
wy50tty1a
```

3. Then the user's start up file must be edited with the appropriate **tset(C)** command line to automatically set the terminal type. In each C-shell user's *.login* file, add the following line:

```
tset -s -Q>/tmp/tset$$;source /tmp/tset$$;/bin/rm /tmp/tset$$
```

Be sure to remove the default **setenv(C)** command line involving **TERM** and **TERMCAP** from the *.login* file.

In each Bourne Shell user's *.profile*, add the following line:

```
eval `tset -s`
```

Be sure to remove the existing **tset** command line from the *.profile* file.

4. Have each user log out, then log in again to test the new terminal type change. After they log in, have them verify the new **termtype** by entering:

```
env
```

14.10 Removing a Terminal

From time to time it may be necessary to remove a terminal from the system, for example, if you wish to replace it with some other device. Before you can remove a terminal, you must disable it with the **disable(C)** command.

To remove a terminal, follow these steps:

1. Turn off the power to the terminal.
2. Log in as the super-user at another terminal.
3. Use the **disable** command to disable the terminal. The command has the form:

```
disable ttyname
```

where *special file* is the name of the serial line to which the terminal is attached. For example, the command:

```
disable /dev/tty1a
```

disables the terminal connected to serial line */dev/tty1a*.

4. Disconnect the terminal from the system.

The serial line previously connected to the terminal is now free to accept another device.

When using the **disable** command, make sure that you wait a full minute between each use of the command. Failure to do so can cause a system crash.

14.11 Modem Usage under XENIX



14.11.1 Serial Lines

XENIX supports modem control on serial ports. The following device names refer to the serial ports with and without modem control.

Device: Function:

<i>/dev/tty1a</i>	main serial adapter without modem control.
<i>/dev/tty1A</i>	main serial adapter with modem control.
<i>/dev/tty2a</i>	alternate serial adapter without modem control.
<i>/dev/tty2A</i>	alternate serial adapter with modem control.

/dev/tty1a and */dev/tty1A* refer to the same serial port (likewise for */dev/tty2a* and */dev/tty2A*). The operating system uses different device-driver subroutines for each. Never attempt to use both modem and non-modem control ports at the same time or you will see the warning:

```
cannot open: device busy
```

For systems including multi-port serial cards, the devices */dev/tty[1,2][a-m]* refer to use *without* modem control, and the devices */dev/tty[1,2][A-M]* refer to use *with* modem control.

14.11.2 Dialing Out From Your Computer

The **cu**(C) and **uucp**(C) utilities are used to call remote systems and transfer data under XENIX. The file */usr/lib/uucp/Devices* (referred to as *Devices*) contains information used by these programs to determine the characteristics of a particular serial line.

The *Devices* file contains lines which specify the device for the line, the call-unit associated with the line, and the baud rate, that are to be used by UUCP. (Modem control devices should be used with lines connected to modems.)

Using Dialer Programs

For dialing, both **cu** and UUCP use a common set of dialers, which can be standalone binaries (programs) like */usr/lib/uucp/dialHA12*, or entries from the file */usr/lib/uucp/Dialers*. (For more information on *Dialers* file entries, see the “Building a Remote Network with UUCP” chapter in this guide.)

The source for several dialer programs and a makefile for recompiling the source program are included in the directory */usr/lib/uucp*. If you have any other kind of modem, you can modify any of the source files and create your own dialer program. Note that you must have the XENIX *Development System* installed to compile a program.

To make a new dial program, follow these steps:

1. Change directory to */usr/lib/uucp* with the following command:

```
cd /usr/lib/uucp
```

2. Edit the file *makefile* in the directory */usr/lib/uucp* and find the line that reads:

```
EXES= dialHA12 dialHA24 dialTBIT dialVA3450
```

and add the name of the dialer program you wish to use. When this is done, exit the file, saving the changes you have made.

3. Next, enter the command:

```
make
```

to your shell prompt and press **RETURN**.

4. When the **make** command is finished, you have a new dialer program. This can be used in the fifth field of an entry in the *Devices* field.

When you are hooking up your modem, or any other device, make sure that serial wires connected to your computer are not left hanging. An unterminated line connected to your computer can considerably reduce system performance. Unplug a modem wire at the computer. Three-wire cables are not sufficient when using modems. Several other pins must be connected for the modem to operate properly. If you are unsure as to what to use, a ribbon cable that connects all pins will work correctly.

14.11.3 Installing a Dial-out Modem

1. Make sure the UUCP package is installed. Use **custom(ADM)** to install if necessary.
2. Make sure the serial port you have chosen for your dial-out modem is recognized at boot up and, if the modem is internal, make sure the COM port the internal modem is configured for does not conflict with any other device. Only serial devices attached to COM1 and COM2 are supported.
3. Make sure the port is disabled by entering:

disable *ttyname*

4. Connect the modem to the machine using a straight-through cable (pins 2 & 3 not crossed). The cable must have at least pins 2, 3, 7, 8, and 20 connected.

Most standard COM boards use straight through cables, but some hardware requires a null-modem cable (pins 2 & 3 crossed). A standard COM board is known as DTE, a board that needs a null-modem cable is known as DCE. Check your hardware documentation if you are not sure. If the COM board is a DCE, you will need a null-modem cable.

5. Add the correct entry to the */usr/lib/uucp/Devices* file. This file has one line for each different dial-out port.

6. Enter the following command to establish UUCP as the owner of the port you have selected:

```
chown uucp /dev/ttyname
```

7. Test the dial-out modem. To test the modem's ability to dial correctly use the following command:

```
cu -s1200 -l ttyname dir
```

where *ttyname* is the device name of the modem control port where you attached the modem. You should see a message indicating that you are connected. If you do not see such a message, either the **cu** command is incorrect, the *Devices* file is incorrect or the serial port is not operating correctly.

Note

The instructions that follow assume a Hayes-compatible command set and response codes. Other modems may use other conventions.

14

If you do see a message confirming your connection, enter

```
AT
```

on your keyboard. "OK" should be echoed back onto your computer screen. If the modem is set to return result codes as numeric codes rather than text, you will see a 0. If this does not occur, check that the "send" light on the modem flashes when you press a key. This indicates the modem is receiving signals from the keyboard. If this light is not flashing, check your cable and modem switch settings. If the "send" light flashes, but you still don't get an "OK" response from the modem, try entering

```
ATE1
```

at your computer keyboard to enable the modem's echo capability.

If you get the expected responses, you can dial out by entering

```
ATDT phonenumber
```

or you can exit **cu** by entering:

and then press **RETURN**.

8. You are now ready to dial another XENIX/UNIX system. If you have any problems, refer to the next section on troubleshooting your dial out modem. If the line is also to be used for dial-in, follow the additional steps specified in "Installing a Dial-in Modem" in this chapter.

14.11.4 Troubleshooting Your Dial-out Modem

The examples below assume a modem attached directly to COM1. Other serial ports are often used. If you have problems, first verify that the phone jack is plugged in and you have a dial tone on the phone line.

14

1. **Problem:** In testing the modem connection with the command:

```
cu -s1200 -l tty1A dir
```

I get a connected message but when I type "AT" there is no "OK" message.

Remedy A: Check the cable and modem switch/software settings. If using a straight through cable try a null modem cable using at least pins 2, 3, 7, 8 and 20. If the "send" light flashes, try using ATE1 to turn on the modem's echo capability.

Remedy B: The modem may be defective. If this is the case, check your hardware documentation for an appropriate repair facility.

2. **Problem:** Modem dials but call never connects.

Remedy A: The phone number may be incorrect or not operational. The phone line to which the modem is attached may be faulty.

Try dialing the number from a telephone to make sure you get a modem on the other end of the line. Use the same phone line you are using for the modem.

Remedy B: Try adding pauses to your telephone number. A hyphen is used to indicate a pause of two seconds, for example: "9---458--1234".

Listen carefully to the calling sequence to make sure the modem is pausing at the right moments. Check your hardware documentation to assure that you are using the right code to indicate a pause.

3. **Problem:** In dialing out you see the message:

```
Connect failed: NO DEVICES AVAILABLE
```

Remedy A: There is no entry in the *Devices* file for the modem port.

Remedy B: The modem port in *Devices* does not have the correct baud rate associated with it.

4. **Problem:** Modem answers but I get garbage characters on my terminal.

Remedy A: The remote computer is at a different baud rate. If you are dialing in to another XENIX system, send a break signal to cause the remote site to switch baud rates during the login sequence. Always have XENIX start at the highest baud rate and move down as necessary. Enter:

```
^%b
```

Remedy B: The site you are calling may be set to different data bit and parity values than you are using. By default **cu** uses 8 data bits, and no parity. Use **cu -e** for 7 data bits, even parity, and **cu -o** for 7 data bits, odd parity.

Remedy C: There may be noise on the line. This becomes more acute when operating at 2400 baud or higher. Check your phone line.

5. **Problem:** My modem does not hang up at the end of a call.

Remedy A: A non-modem control port is being used. Change the non-modem control serial port you specify to the corresponding modem control port. For example, the modem control port associated with `tty1a` is `tty1A`.

Remedy B: The modem is not lowering the CD line when the call is disconnected. Check the modem switches to verify that the modem is set to follow the incoming carrier or, if this is a Hayes 2400 or compatible modem, use the `AT&C1` command.

Remedy C: The modem is not set to watch DTR. Check the modem switches to verify that the modem is set to watch DTR or, if this is a Hayes 2400 modem, use the `AT&D2` command.



14.11.5 Dialing Into Your Computer Under XENIX

To allow dialing into your computer, you must enable a serial line that recognizes modem control signals, with the **enable(C)** command. When using the **enable** command, make sure that you wait a full minute between each use of the command. Failure to do so may send too many signals to the **init(C)** program, which will then terminate. If **init** terminates, no new logins are possible.

To use the main serial adapter, enter:

```
disable tty1a  
enable tty1A
```

Or, for the the alternate serial adapter, enter:

```
disable tty2a  
enable tty2A
```

Note that **tty1A** and **tty1a** refer to the same (main) serial line, and **tty2A** and **tty2a** refer to the same (alternate) serial line. Do not use the same line in both its modem and non-modem modes at the same time as this will cause an error.

14.11.6 Installing a Dial-in Modem

The following procedure provides step by step instructions for installing a modem for dial-in operation. (Passwords are recommended for dial-in lines; refer to the section on “Adding Dial-in Password Protection” in the “Maintaining System Security” chapter for details.)

1. Follow the steps for installing a modem for dial-out. This ensures that you have a working hardware connection.
2. Some modems have switches or software commands for setting the modem configuration. If your modem has such settings, configure it as follows, following the instructions in your modem manual.

Note

If the modem is to be shared between dial-in and dial-out, the next step can be omitted. Initialization to dial-in is performed whenever the system comes up, or a dial-out completes (**getty** runs the dialer program with the **-h** option, or uses an **&chat** entry from the *Dialers* file, to reinitialize the modem.

3. Set the modem to automatically answer the phone when a call comes in.

Most internal modems do not have auto-answer and some external modems do not have this setting. If this is the case, place the following line in your */etc/rc* file.

```
(stty 1200; echo "ATS0=1\r" >/dev/tty1a) </dev/tty1a
```

“*tty1a*” should be changed to match the non-modem control device the modem is connected to. “1200” should be changed to the highest baud rate used by the modem. “ATS0=1” is the command to put Hayes compatible modems into autoanswer mode. The “\r” is needed to send a carriage return signal to the modem to terminate the command line.

4. Set up your modem so that it does not answer when the DTR line is not active and it disconnects from the current connection when DTR goes from active to inactive.
5. The CD line should follow the incoming carrier, i.e. low when a carrier is present, high when a carrier is not present.
6. Set up your modem so that it does not echo commands or display responses.
7. Make sure the port is disabled by entering the command:

```
disable ityname
```

Where *ityname* is the non-modem control port.

8. Select the desired **gettydefs** entry in the */etc/ttys* file. Entry “2” will select the 1200-2400-300 cycle.



9. Enable the port you are using for your modem with the following the command:

enable *ttyname*

Where *ttyname* is the modem control port.

10. Dial this modem from another modem.
11. If you are unable to successfully dial-in, see the next section on trouble shooting your dial-in modem.

14.11.7 Troubleshooting Your Dial-in Modem

The examples below assume that your modem is attached directly to the COM1 port. In practice, a modem may be attached to other serial ports.

1. **Problem:** Modem is not answering phone.

Remedy A: The modem serial port is not enabled.

Enter the following command:

enable /dev/tty1A

Remedy B: The modem is not configured to auto-answer.

Check your modem switches or, if this is a Hayes 2400 modem, use the appropriate modem software command. Enter "**cu -l tty1A dir**" to the modem and use the command "**ATS0=1**" to set auto-answer.

Remedy C: The DTR line is not connected from the computer to the modem.

Check Pin 20 and make sure it is connected. Pins 2, 3, 7, 8, and 20 are used for modem hookup.

2. **Problem:** Modem answers, but hangs up immediately upon connection.

Remedy: The modem is set to autoanswer and to watch the DTR line, but the DTR line is not asserted. Check the following possibilities:

a) The modem control port may not be enabled. Enter the command:

```
enable /dev/tty1A
```

b) The cable is incorrect.

If you are using a straight through cable with at least pins 2, 3, 7, 8 and 20 connected, check that pin 20 (DTR) is properly connected.

3. **Problem:** I see the error message “Garbage or loose cable on /dev/tty1A, port shut down” on my console when a call comes in to my modem.

Remedy: Your modem is set to echo back data or send responses to commands.

It is very likely that the modem is sending a “RING” signal to indicate that the phone you are calling is ringing. Since the CD signal is not active, getty interprets this as random data on the serial line. To correct this, set the modem to turn off echo and not to send command responses. The proper Hayes 2400 modem command is “ATE0Q1”.

4. **Problem:** The modem answers, but I get no login prompt.

Remedy A: The CD line is not being asserted by the modem after the modem has answered the phone.

Check the switches on your modem or, if this is a Hayes 2400 modem, use the appropriate modem software command.

Remedy B: The port is not enabled. Enter the command:

```
enable /dev/tty1A
```

Remedy C: An incorrect */etc/gettydefs* entry is being used and is selecting the wrong baud rate.

Check the modem port device in the */etc/ttys* file. It will appear similar to the following example:

```
12tty1A
```

The first digit is a 0 or 1 depending on whether the device is disabled or enabled. A 0 means the device is disabled, and a 1 means the device is enabled. The second character is a pointer to an entry in the */etc/gettydefs* file. Check the baud rate and serial line characteristics of this entry.

5. **Problem:** The screen scrolls when I log in.

Remedy: The modem and non-modem devices are both enabled.

Disable the non-modem device by entering the command:

```
disable /dev/tty1a
```

6. **Problem:** I get a login prompt but nothing coherent afterward.

Remedy: The line settings are incorrect.

Check the */etc/ttys* file to verify that the “pointer” into the *gettydefs* file is correct. Chances are that the serial line characteristics don't match between the *stty* settings defined in the third field of the selected *gettydefs* entry. Try changing the terminal's setup to 8 data bits, one stop bit, and no parity.

14

14.11.8 Shared Dial-in/Dial-out

XENIX supports the use of dial-in and dial-out on the same modem line, without having to disable the login.

When a dial-out program is using the line, the login will be disabled. If someone is logged in on a line when a dial-out program attempts to use it, the dial-out program will fail to lock the device.

For this feature to work correctly, the modem control device must be used, and the modem must set CD to high when a carrier is present and low when a carrier is not present. (For information on using dial-in/dial-out in conjunction with **uucp**, refer to the “Building a Remote Network with UUCP” chapter in this guide.)

14.11.9 Installing A Shared Dial-in/Dial-out Modem

The following procedure allows you to install a shared dial-in/dial-out modem.

1. Perform the steps of installing a modem for dial-out, and those for installing a dial-in.
2. To dial out, simply invoke **cu**. The *getty* will automatically be suspended for the call out, and restarted when the call is completed.

14.11.10 Hayes Modem Settings

Proper modem configuration is necessary when using **cu** and **uucp**. Modem settings differ for each modem. Consult your modem manual for the proper switch settings.

Smartmodem 1200

If you have a Hayes Smartmodem 1200 or compatible, switches 3 and 8 should be down:

	1	2	3	4	5	6	7	8
up	•	•		•	•	•	•	
down			•					•

When switch 3 is down, the resulting codes will be sent to, (echoed by), the modem to the terminal or computer. When switch 8 is down, the modem is able to interpret the command being issued. This allows both the XENIX and DOS communications systems to work.

14

Smartmodem 2400

The Hayes 2400 Smartmodem or compatible modem requires on-line configuration if it is to be used as a dial-in line. Note that the Hayes 2400 does not answer the phone with a 2400 baud carrier if it is not set up with 2400 baud commands. You must configure the modem by issuing set up commands via **cu(C)**. The form of the **cu** command is:

```
cu -s2400 -l tty $nn$  dir
```

nn is the "tty" number of the serial line. To configure a modem on tty1A, enter this command and press **RETURN**:

```
cu -s2400 -l tty1A dir
```

Next, enter the following commands to configure the modem. They will be saved in the modem's non-volatile memory. If you do not want to save the settings, do not enter the last command (**at&w**). Commands are in the left column and short descriptions of what they do are in the right column. Follow each command with a **RETURN**:

- **AT&f** Fetch factory configuration.
- ATT** Tone dialing.
- ATI0** Low speaker volume.
- AT&d2** Set dtr "2": go onhook when dtr drops.
- AT&c1** Set dcd "1": dcd tracks remote carrier.
- ATs0=1** Answer phone after 1 ring (AA light should come on).
- **ATs2=128** Disable modem escape sequence.
- ATe0** No echo (modem will no longer echo what is sent to it).
- ATq1** Quiet mode (modem will not respond with "OK" after this command or any that follow).
- AT&w** Saves settings in non-volatile memory.

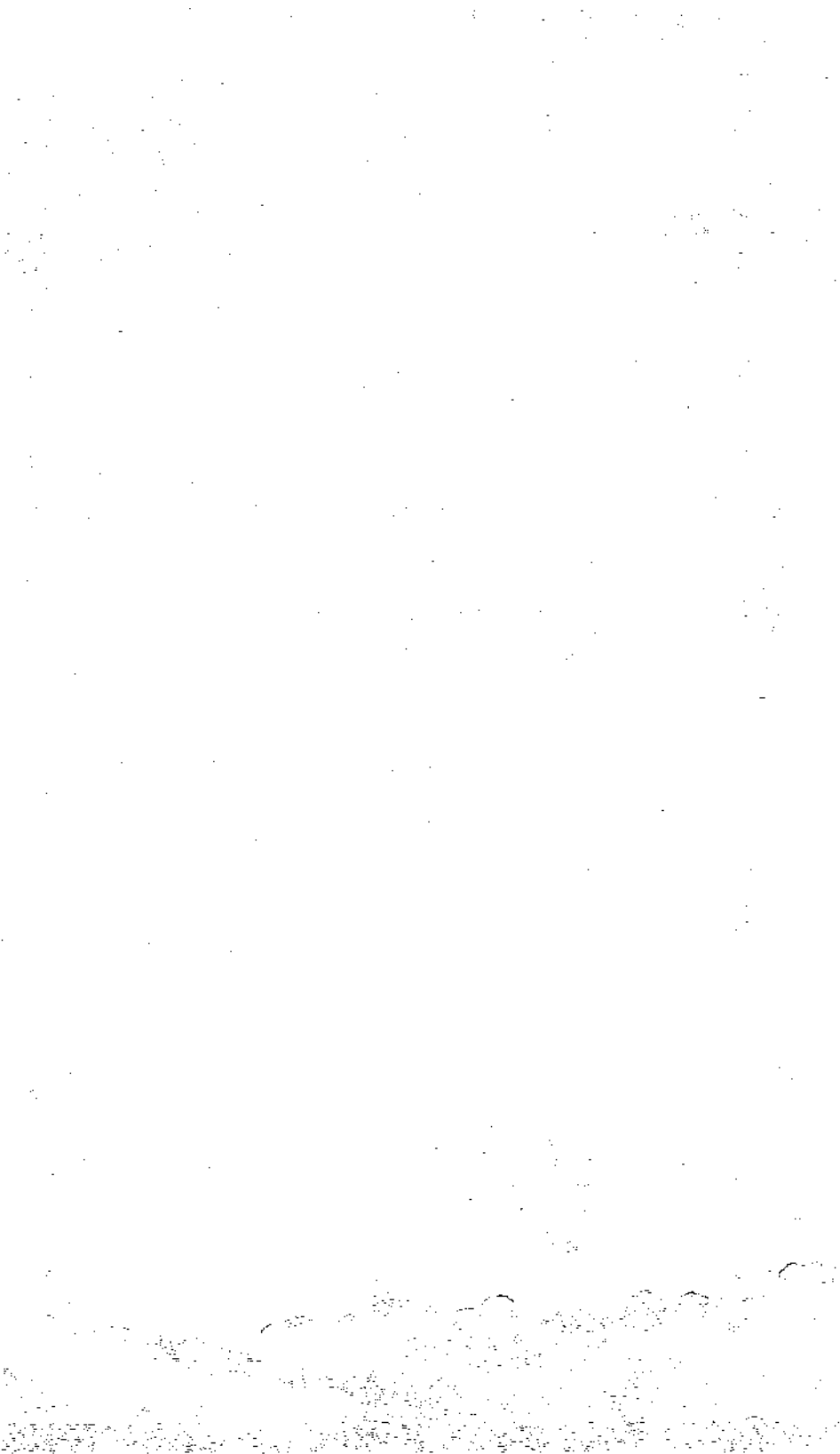
Exit from **cu** by entering a "tilde" and a "period", followed by a **RETURN**:

The modem is now configured and ready for use.

Chapter 15

Using Printers

- 15.1 Introduction 15-1
- 15.2 The Printer Spooling System 15-1
- 15.3 Installing a Printer 15-2
- 15.4 Stopping the Print Spooling Daemon: lpsched 15-8
- 15.5 Creating an Init Device File 15-9
- 15.6 Moving Requests Between Printers: lpmove 15-10
- 15.7 Controlling Print Requests: accept 15-11
- 15.8 Canceling a Print Request 15-12
- 15.9 Enabling and Disabling Printers 15-12
- 15.10 Printer Interface Programs 15-13
- 15.11 Adding a Local Printer 15-15



15.1 Introduction

Printers are a highly important attachment to any computer system. Most systems require the ability to print out data on paper. XENIX supports a wide variety of printing hardware. Some lineprinters are *parallel* devices, but most will be connected as *serial* devices.

To add a printer, the system administrator must:

- Connect the physical hardware to the computer, then
- Use the correct system commands to enable the printer for operation.

This chapter explains how to do this and how to maintain printers once they are added.

Note that physical connections between a printer and the system vary depending on hardware configuration. This chapter provides some information about making the necessary physical connections, but for more information about these connections, see the hardware manuals provided with the printer and your computer.

XENIX supports serial printers that use the standard RS-232 interface. To find out if your printer uses this interface, check your hardware documentation. XENIX also supports RTS/CTS protocols.

15

15.2 The Printer Spooling System

The XENIX lineprinter spooling system is a collection of commands that help you, as system administrator, to efficiently install, monitor, and control the lineprinters serving your system. A request to print a file is “spooled” or queued with other printing jobs to be sent to the printer. Each print job is processed and waits its turn in line to be printed, thus the term “queue.” When a user requests a file to be printed using the **lp(C)** command, the lineprinter system responds with a “request ID.” This consists of the name of the printer on which the file will be printed and a unique number identifying the file. With this request ID, the user can find out the status of the print request or cancel it. The **lp** options help the user to easily control printer output.

There are several terms used to describe the line printer system:

<i>device</i>	The target for lp output. It can be a hard-wired printer, a terminal that is sometimes used as a printer, or a regular file. A device can be represented by a full XENIX path-name.
<i>printer</i>	The name assigned by the system administrator to represent a device. This name can have up to 14 characters. At different times, a printer may be associated with different devices.
<i>class</i>	An ordered list of printers. Print requests sent to a class of printers are printed by the first available member of that class.
<i>destination</i>	A <i>destination</i> is where print requests are sent. A destination can be a class or a printer.



Consult your computer and lineprinter hardware manuals for information on making the connection between your system and printing devices.

15.3 Installing a Printer

This section instructs you how to install new printing devices on your XENIX system. You must connect the printer to a proper port, (serial port for serial printers, parallel port for parallel printers) ensure that it works, and set up the XENIX printer spooling software using the **mkdev lp** command. Follow the steps below to install a printer:

1. Find a place for your printer and make sure that it is properly assembled and plugged in to a power outlet.
2. **If you are connecting a serial printer:** connect the RS-232 cable from your computer serial port to the port on your printer. Serial printers must be capable of supporting XON/XOFF or DTR protocols and must be configured for those protocols. Next, enter the following command substituting the correct port number for *nn*:

disable /dev/ttynn

and press **RETURN**. This disables logins on the port you have connected to your printer and allows the port to be used for serial communication.

3. **If you are connecting a parallel printer:** The printer must use a standard Centronics interface cable. The parallel port on a monochrome card should be configured for interrupt vector 7, and is recognized as **lp1** when booting up. The main parallel port should be configured for interrupt vector 7 and is recognized as **lp0**. So you must use either the main or the monochrome port - not both - to avoid a hardware conflict. The alternate/second parallel port should be configured for interrupt vector 5, and recognized as **lp2**. Make sure no other hardware is using these interrupts. (See your hardware manual for information on configuring your parallel ports.)
4. Verify that you have correctly hooked up the printer by sending data directly to the device. Enter one of the following commands:

For serial printers:

```
date > /dev/tty $nn$ 
```

where nn is the number of the serial port you are using (for example, */dev/tty1a*).

For parallel printers:

```
date > /dev/lp $n$ 
```

where n is the number of the parallel port you are using (for example */dev/lp0*).

5. If you do not see the date printed on your printer, there is most likely some type of hardware malfunction, so verify the following:

For parallel printers:


- Make certain your cable is securely connected and all wires are good. Using the cable on a known good system, or printing under DOS are good ways to test this.
- Re-check your printer configuration by verifying its switches in your printer hardware manual.

- Re-check the switches on your parallel card. It must also be recognized at bootup. You can verify this by rebooting your system and looking for the following message:

```
Parallel port lpn present
```

For serial printers:

- Make certain you are using the non-modem control device, for example: `/dev/ttyla`, not `/dev/ttylA`. (For more information on the naming convention for serial ports, see **serial(HW)** in this guide.) Try using a cable with only pins 2, 3 and 7 connected.
- Re-check your printer configuration by verifying its switches in your printer hardware manual.
- Re-check your switches on your serial port. If you are using a multi-port card, try other lines on that card and be sure it does not conflict with the standard COM ports.
- Try attaching the printer to a standard serial port, COM1 or COM2, to see if the printer and cabling is correct.

- 
6. Once you have the printer correctly hooked up and responding, you are ready to run **mkdev lp**. You must know the port you are using or the XENIX pathname of the device (for example, `/dev/ttyla`) and the lineprinter interface program you are going to use. A model interface program is supplied with your XENIX system. For more information on printer interface programs, see "Printer Interface Programs."
 7. **mkdev lp** prompts you with a series of questions. For most of them you need to supply your own answers, entering the information as you are prompted. When you are prompted for the printer name, however, you are given a default name option. If you wish to choose the default answer, simply press the **RETURN** key. If you make a mistake while responding to the questions, just press the **DELETE** key or the **INTERRUPT** key and start again.

Enter the following command:

mkdev lp

△ **sysadmsh** users select: System→Add→Printer

8. If the scheduler is running you are reminded that any jobs which are printing may be interrupted and you are prompted if you want to continue.
9. The following menu is displayed:

```

Do you wish to:
  1 - Add a new printer.
  2 - Remove a printer.
  3 - Reconfigure an existing printer.
  4 - Assign a default printer.
  5 - Print lp status information.

Select an option or enter q to quit:

```

Enter **1** and press **RETURN**.

10. Next you see the following menu:

```

1. Parallel Printer
2. Serial Printer
3. Remote Printer

Select the type of printer you are adding
or enter q to return to the previous menu:

```

11. A list of available parallel or serial devices is displayed.

Enter your choice and press **RETURN**.

12. The following message is displayed:

```

Enter a name for the printer, press <RETURN>
to use the default name (printer) or enter q to
return to the previous menu.

```

Printer names can be up to 14 characters long and can be any combination of numbers, letters, or underscore characters. Enter the printer name and press the **RETURN** key.

- 13. If you specified that you are adding a remote printer, you now see the following messages and you are prompted to answer questions about the remote printer. If you specified Serial or Parallel you will not see these messages. For remote printers, you see:

```
Enter the node name of the remote computer to which your printer
is connected or enter q to return to the previous menu:
```

Next you see:

```
Enter the name of the printer you wish to use on node_name
or enter q to return to the previous menu:
```



Next you see the message:

```
Is node_name connected via (M)icnet or (U)ucp?
```

Enter "M" or "U" depending on the network your system uses.
Next you see:

```
Is your remote printer an Imagen laser printer?
```

Once you have answered these questions for your remote printer, you see the "Printer enabled and accepting request" message described below. You do not see the other messages described here for serial and parallel printers. Note that this is to set up printing from a remote machine. You must also run **mkdev lp** for the printer locally on the machine where it is physically connected.

14. The following message is displayed:

```
If you have an unusual printer you must create an
interface program in /usr/spool/lp/model. For a sample
interface program look at /usr/spool/lp/model/dumb.
```

15. Now you are prompted for a printer type. The screen displays a numbered list of the available printer types on your system, followed by these instructions:

```
Enter one of the interface programs above or the
full pathname of an interface program or enter q
to return to the previous menu:
```

Enter one of the options, then press **RETURN**.

16. After you have finished responding to these questions, the following message is displayed:

```
destination printername now accepting requests
printer printername now enabled
```

printername is then automatically enabled and is ready accept print requests.

17. After enabling the printer, you are prompted to respond to one more question:

```
Is this the default printer ? (y/n)
```

You can enter **y** (for yes) or **n** (for no) depending upon whether you want user print requests to be automatically routed to the printer.

18. One last message is displayed:

```
If you need to modify your interface program
after installing it, the installed version is
in /usr/spool/lp/interface/printername.
```

After you have responded to these questions, **lpsched**(ADM) is automatically started, and users can begin printing files on the new printer.

You can also add printers to your system using the **lpadmin**(ADM) command. However, you will need to give separate commands to stop **lpsched**, to enable the printer, and to allow it to accept print requests. For more information on these programs and commands, see sections "Stopping the Print Spooling Daemon," "Controlling Print Requests," and "Enabling and Disabling Printers."

15.4 Stopping the Print Spooling Daemon: lpsched

15

The spooling daemon, **lpsched**, routes print requests through the correct printer interface program and then to the lineprinter. No printing can be done on your system unless **lpsched** is running. The program **lpsched** starts automatically each time your XENIX system is restarted. Sometimes it is necessary to stop **lpsched**, especially if you want to reconfigure printers or if you want to add new printers using the **lpadmin** command (**mkdev lp** automatically stops and restarts **lpsched**).

This section explains how to find out whether or not **lpsched** is running, how to stop and restart it, and how to recreate it if necessary.

To find out whether or not **lpsched** is running, you use the **lpstat**(C) command:

```
lpstat -r
```

Δ **sysadmsh** users select: System→Report→Printers

The system responds with a message indicating whether **lpsched** is either running or not.

To shutdown the scheduler, **lpsched**, enter:

```
/usr/lib/lpshut
```

lpsched stops running and all printing stops as well. Printing requests stopped in the middle of printing reprint when **lpsched** starts again.

After you have finished configuring the printers, you should restart **lpsched**. To do this, enter:

```
/usr/lib/lpsched
```

lpstat -r should confirm that **lpsched** is running.

Each time **lpsched** sends a print request to an interface program, it records an entry in a log file, */usr/spool/lp/log*. The entry includes the user name, the request ID, the name of the printer the request will be printed on, and the date and time requested. **lpsched** also records any error messages in this file. After you have stopped **lpsched**, the log file is renamed */usr/spool/lp/oldlog* and **lpsched** starts a new log file. Requests waiting to be printed before **lpsched** was stopped may have an entry in both log files.

For more information on **lpsched**, see **lpsched(ADM)** in this guide.



15.5 Creating an Init Device File

The standard parallel printer devices (*/dev/lp*, */dev/lp0*, */dev/lp1*, and */dev/lp2*) send a printer initialization string (*init*) the first time the device is opened after the system starts up. This is done on the first open only so that printers with large text buffers will not be flushed by the sending of another file.

Some parallel printers require initialization every time a file is received for printing. Others require an *init* if the printer is turned off and back on again (for example, after changing paper or ribbons). The symptom of this situation is that the printer works fine until it is turned off and then back on.

If you need to initialize the printer more often than the standard devices provide, you can create an additional device file for the parallel port in use. This “init device file” can be used when necessary to initialize the printer.

1. Log in as superuser.
2. Determine which device is the parallel port you are using. This example assumes the device is the main parallel port (*/dev/lp0*).

3. Use **mkdev lp** to associate one of the parallel *init* devices (*lp0i*, *lp1i*, *lp2i*) with the printer (select option 3: "reconfiguring an existing printer").

If your printer needs an *init* when it is turned off and on, use the following command line after the printer is turned on. Initialize the printer before the first file is sent to the printer (this example assumes the main parallel port):

```
>/dev/lp0i
```

If your printer needs an *init* every time a file is sent (and it doesn't have a large internal text buffer) you can use the */dev/lp0i* device all the time.

The **lp(C)** command now sends an **init** every time a file is sent to the printer.

Some printers do not have newline/carriage return mapping. If your printer needs to have newlines mapped to newline/carriage returns, specify the *crnlmap* filter when you set up the printer using the **mkdev lp** command. When you are prompted for the type of printer (dumb printer, Imagen laser printer (parallel interface), Imagen laser printer (serial interface), or other), enter "other". You are then prompted for the pathname of the interface program. The printer filter file is found in */usr/spool/lp/model/crnlmap*.

15

15.6 Moving Requests Between Printers: **lpmove**

You can move print requests between printing destinations by using the command **lpmove**. This command does not move print requests while the scheduler, **lpsched**, is running. To stop **lpsched**, see "Stopping the Print Spooling Daemon." **lpmove** will move individual print requests by request ID, or all requests waiting to be printed on a particular printer.

For example, to move a request with a request ID of *quick-532* to a printer named *slow*, enter:

```
/usr/lib/lpmove quick-532 slow
```

The print request now has a new request ID: *slow-532*.

To move all requests on a printer named *quick* to *slow*, enter:

```
/usr/lib/lpmove quick slow
```

For more information on **lpmove**, see **lpsched** (ADM), in this guide.

15.7 Controlling Print Requests: **accept**

The **accept** command allows printers or classes of printers to accept print requests made with the **lp** command. You can allow a printer to accept requests after it has been properly configured. The printer, however, will not begin printing the requests until the **enable** command is given. If you added a printer to your system using the **mkdev lp** command, these steps were automatically performed. For information on **enable**, see “Enabling and Disabling Printers.”

For example, to have print requests accepted for a class of printers named *class1*, enter:

```
/usr/lib/accept class1
```

If you want to prevent requests from being routed to a printer, you can use the **reject** command. The **-r** options allows you to send users a message explaining why a printer is out of service.

For example, to prevent printing requests, from being routed to a printer, *printer4*, because of repairs, enter:

```
/usr/lib/reject -r"printer4 needs repair" printer4
```

A user who requests a file to be printed on *printer4* will receive the following message:

```
lp: can not accept requests for destination "printer4"  
--printer printer4 needs repair
```

To find out the acceptance status of printing destinations, enter:

```
lpstat -a
```

For more information on **lpstat**, see “Finding Out the Status of a Print Request” in the “Working with Files and Directories” chapter of the *XENIX Tutorial*. For more information on **accept/reject**, see **accept(C)**, in the *XENIX User's Reference*.

15.8 Canceling a Print Request

To cancel a printout you have requested, use the **cancel(C)** command. When you request a printout, XENIX responds and displays a request ID for your job. For example, if you send a job to printer "laser" on your system, XENIX will display the request ID as:

```
request id is laser-number
```

where *number* is the number assigned to your job. To cancel the job before it begins printing, use the following command:

```
cancel laser-number
```

and the printout will be canceled.

Most systems print quickly, so a **cancel** command must be used promptly to have any effect.

15

15.9 Enabling and Disabling Printers

The **enable** command allows **lpsched** to print files on printers. A printer can accept requests for printing after the **accept** command is given for it, but in order for the files to be printed, the **enable** command must be given as well.

For example, to enable a printer named *daisy*, enter:

```
enable daisy
```

You can disable printers with the **disable** command. The scheduler, **lpsched**, will not send printing requests to disabled printers regardless of their acceptance status. The **-r** options allows you to send a message to users explaining why a printer has been disabled.

For example, to disable a printer named "laser" because of a paper jam, enter:

```
disable -r"paper jam" laser
```

Users requesting the status of “laser” with the command **lpstat -plaser** will receive the following message:

```
printer laser disabled since Dec 5 10:15
paper jam
```

For more information on these two commands, see **enable(C)** and **disable(C)** in the *XENIX User's Reference*.

15.10 Printer Interface Programs

Each printer on your system must have a printer interface program. This can be a shell script, C program, or any other executable program. Your XENIX system provides a model interface program. It is written as a shell script and can be found in */usr/spool/lp/model*. You can use this program as is, modify it, or write your own interface program.

If you want to write or modify a printer interface program, the following information may be helpful.

When **lpsched** routes a printing request to a printer *P*, */usr/spool/lp* invokes the interface program for *P* as follows:

```
interface P id user title copies options file
```

These flags are:

<i>interface</i>	the directory which contains executable copies of interface programs
<i>P</i>	the interface program being executed
<i>id</i>	the request id returned by lp
<i>user</i>	the login name of the user who made the request

XENIX System Administrator's Guide

<i>title</i>	an optional title given by the user
<i>copies</i>	the number of copies requested
<i>options</i>	a list of printer dependent options separated by blanks
<i>file</i>	the full pathname of a file to be printed

When the interface program is started, its standard input comes from */dev/null* and both standard output and standard error output are directed to the printer's device. Devices are opened for reading as well as writing when file modes permit. If a device is a regular file, all output is appended to the end of that file.

Interface programs may format their output in any way. They must, however, ensure proper **stty** modes for terminal characteristics such as baud rate and output options. In a shell script interface, this means the printer device must be open for reading - take the standard input for the **stty** command from the device.

The file */etc/default/lpd* contains a line "BANNERS=*d*" where *d* is the number of banner pages to be printed at the front of every printing request. If *d* is set to 0, no banner pages are printed. Interface programs should examine */etc/default/lpd* and behave accordingly.

After printing is completed, the interface program should exit with a code showing that the print job was successful. Exit codes are interpreted by the printer scheduler, **lpsched**, as follows:

Exit Code	Meaning to lpsched
0	Print job was successful
1 to 127	lpsched found a problem while printing this particular request, for example, too many unprintable characters. This problem will not affect future printing requests. lpsched notifies users by mail that there was an error in printing the request.
greater than 127	These codes are reserved for internal use by lpsched . Interface programs must not exit with codes in this range.

Finally, when problems occur in printing that are likely to affect future

printing requests, the printer interface program should disable printers so that print requests are not lost. When a busy printer is disabled, the interface program will be terminated with a signal 15 so that print requests are not lost.

For more information on printer interface programs, see **lpadmin**(ADM), in this guide.

15.11 Adding a Local Printer

XENIX supports the use of local printers attached to the AUX or PRINT port on the back of a normal serial terminal. These printers are connected via standard RS-232 connections and can significantly reduce the load on shared system resources. The **lprint**(C) command is used to print files on a local printer, but the terminal must be properly configured for the command to work. To add a printer connected to the AUX or PRINT port on the back of a normal printer and use it for local printing, follow this procedure:

1. Connect your local serial printer to the AUX port on your terminal with a standard RS-232 cable with pins 1-8 and 20 connected. Make sure the printer is powered on and on-line.
2. Log in to XENIX on the terminal and verify that the terminal is working correctly.
3. Make sure that the AUX port on your terminal is configured with the same settings as your printer. (baud rate, parity, data bits, xon/xoff, etc.)
4. In order for the **lprint** command to work, **lprint** needs to know how to start and stop local printing for each specific terminal. **lprint** looks in the file */etc/termcap* to find two terminal attributes: PN (start printing) and PS (stop printing). These are escape sequences that must be sent to the terminal to control local printing. Very few terminals have these attributes defined in their **termcap** entries. Use a text editor (such as **vi**(C)) to examine the file */etc/termcap*. (*/etc/termcap* can also be an alternate file, as defined by the **TERMCAP** variable.) Search for the entry for your terminal. For example, if your terminal is a Wyse 60, you would search for "wyse60". The **termcap** entry looks like this:

15

```
w7|wy60|wyse60|WyseWY-60 with 80 column/24 line screen in wy60 mode:\
:is=\E\072\Ee(\BO\Ee6\Ec41\E^4\Ec21\Ed/\
:if=/usr/lib/tabset/std:pt:\
:G1=\EH3:G2=\EH2:G3=\EH1:G4=\EH5:GD=\EH0:GG#0:GH=\EH072:\
:GU=\EH#:GV=\EH6:GR=\EH4:GL=\EH9:GC=\EH8:GF=\EH7:\
:PU=\EJ:PD=\EK:\
:al=\EE:am:bs:bt=\EI:cd=\EY:ce=\ET:cl=\E+:\
:cm=\Ea%i&clR&clC:co#80:dc=\EW:dl=\ER:ei=\Er:im=\Eq:k0=\AI\r:\
:k1=\A@r:k2=\AA\r:k3=\AB\r:k4=\AC\r:k5=\AD\r:k6=\AE\r:k7=\AF\r:\
:k8=\AG\r:k9=\AH\r:kcd=\J:kh=\~:kl=\H:kr=\L:ku=\K:\
:li#24:mi:nd=\L:se=\BG0:so=\BG4:sg#0:ug#0:ue=\BG0:ul:up=\K:us=\BG8:\
w8|wy60w|wyse60w|WyseWY-60 with 132 column/24 line screen in wy60 mode:\
:is=\E\073\Ee(\BO\Ee6\Ec41\E^4\Ec21:\
:if=/usr/lib/tabset/std:pt:\
:G1=\EH3:G2=\EH2:G3=\EH1:G4=\EH5:GD=\EH0:GG#0:GH=\EH072:\
:GU=\EH#:GV=\EH6:GR=\EH4:GL=\EH9:GC=\EH8:GF=\EH7:\
:PU=\EJ:PD=\EK:\
```

The Wyse 60 does not have PN and PS defined. Just as with other terminals, you must add a line containing these two attributes to the */etc/termcap* entry for your terminal. The line you will add has the form:

```
:PN=start sequence:PS=stop sequence:\
```

5. Refer to your terminal manual to find the sequence of control characters used to switch the auxiliary port on and off. This is sometimes referred to as "passthrough" or "transparent" mode. For an example of the sequence to enable auxiliary printing, the code to switch the port on for a Wyse 60 terminal is:

```
ESC d #
```

And the code to turn it off again is:

```
Ctrl-T
```

6. These keystrokes must be translated into **termcap** format before inserting them into the **termcap** file. **termcap** uses the following codes to represent keystrokes:



Keystroke	termcap sequence
ESCAPE	\E
CTRL- <i>x</i>	^ <i>x</i> (<i>x</i> is any character)
NEWLINE	\n
RETURN	\r
TAB	\t
BACKSPACE	\b
FORMFEED	\f

To use a control sequence, use the caret (^) symbol, not the **Ctrl** key. For example, **Ctrl-x** would be represented by ^*x*. In addition, characters can be represented by their octal codes (see **ascii(M)**), and the caret (^) and \ characters represented by \^ and \\, respectively. Entries for **termcap** attributes must be separated by a colon (:). (See **termcap(M)**.) for more details.)

Recall that the **termcap** attributes for starting and ending printing are PN and PS. Using the table above **termcap** entry for the Wyse 60 keystrokes **ESC#d** (start printing, PN) and **Ctrl-T** (stop printing, PS) looks like this:

```
:PN=\Ed#:PS=^T:\
```

- For a Wyse 60 terminal, you would simply insert the above line into the **termcap** entry for the Wyse 60. (You must be certain to insert the line within the entry for your terminal; don't add it as the first line or the last line.) For other terminals, check your owner's manual and locate the proper sequences for turning the auxiliary print mode on and off and substitute the **termcap** sequences as in the example above. Some terminals (such as the Wyse 60) include a "transparent" mode, where the data is not displayed on the screen as it is printed. (This is the mode selected by the PN sequence in the example.)

Note

You must be logged in as **root** to edit */etc/termcap*. We recommend that you copy the original file to another name in case you make an error. You can also extract the file again from your distribution using **custom(ADM)**.

8. Once you have added the PN and PS entries, log out and back in again to activate the new termcap entry.
9. Use the following command will print the file *filename* on your local printer.

lprint *filename*

Do not touch your keyboard while local printing is taking place; you cannot perform other tasks on your terminal while printing.

10. If your file is printed on the screen instead of the printer, the PS and PN entries you created are incorrect. Revise the entries with the correct codes. If the file still does not print on the printer or the terminal, try crossing the Transmit and Receive Data pins in the cable connecting the terminal AUX port and the printer.

Chapter 16

Using Floppy Disks and Tape Drives

- 16.1 Introduction 16-1
- 16.2 Using a Cartridge Tape Drive 16-1
 - 16.2.1 Installation and Configuration 16-1
 - 16.2.2 Installing A Mini Tape Drive 16-2
 - 16.2.3 Using A Tape Drive 16-4
- 16.3 Using Floppy Disks 16-6
 - 16.3.1 Formatting Floppy Disks 16-6
 - 16.3.2 Copying Floppy Disks 16-7
 - 16.3.3 Using Floppies for File Storage 16-9
 - 16.3.4 Making Filesystems on Floppy Disks 16-10
 - 16.3.5 Creating an Emergency Boot Floppy 16-13



16.1 Introduction

An important part of any computer system is the ability to offload files and restore them when needed. There are several types of media used to store and recall files. Among these are floppy disks and magnetic tape devices. This chapter explains how to install and use the above types of storage media with your system. Your system should come with at least a floppy disk drive already installed and ready to run. This chapter provides instructions on how to add tape drives and how to use floppy disks.

16.2 Using a Cartridge Tape Drive

A tape cartridge drive is a mass storage device that uses 1/4 inch tape cartridges to store data. It is also referred to as a QIC (quarter inch cartridge) tape drive. A tape cartridge can hold many times the data that can be stored on floppies, making it much more useful for large backup operations.

The drives that are supported under XENIX are listed in the Operating System *Release Notes*. For hardware-specific information, refer to the manual for your drive and **tape(HW)** in this guide.

16.2.1 Installation and Configuration

Read your tape drive hardware manual for physical installation instructions and general information.

16

You need to know the following technical information before you install your tape drive:

- The interrupt number. The default interrupt is set on your tape drive controller card. If this number conflicts with one that is already in use, you must change the setting on the card. Interrupts 0, 1, and 6 are always used by XENIX, even if no other devices are present. If you set the interrupt to anything other than the default, write down the setting you chose, as you need to specify it when you run the **mkdev** utility.
- The DMA Channel and base I/O address. There are also default settings on your tape drive controller card for DMA channel and base I/O address. If you need to change these because of a conflict with existing hardware, note the settings you select and specify them when you run **mkdev(ADM)**.

Once it is connected to your computer, enter this command to configure the drive:

mkdev tape

mkdev runs an interactive program called **tapeinit** which explains the configuration process and prompts you for necessary input. If you are using the default settings on your controller card, select the option to "Install Cartridge Tape Driver" and then enter **q** at the second menu to use the default tape parameters.

If you modified the default settings on your controller card, first select option 1 to install the tape driver, then select the option to "Modify Current Tape Parameters" at the second menu. Next, you see a menu with the default tape parameters. Change any tape parameters here that you changed on your controller board. Note that if you changed the base address, you must enter an "H" after the number if it is a hexadecimal address. If you do not specify the "H", the system assumes that the address is decimal. Also, note that if you choose interrupt 2 on your controller, you must specify interrupt 25 when you modify your tape parameters. The software interrupt 25 corresponds to the hardware interrupt 2. All other interrupts use the same number in software as in the hardware.

When the tape initialization is done, reboot your system and check to assure that the tape device is recognized at boot time. The system should display a message that a "streaming cartridge tape" is present. See **mkdev(ADM)** for more information on installing tape drives.

16

16.2.2 Installing A Mini Tape Drive

Mini tape drives use the floppy disk drive controller and are significantly different from standard QIC tape drives. For one example, mini tapes must be formatted before they can be used. There are also some differences in the installation of mini tapes.

First make sure that your drive is correctly jumpered. The correct setting may be different for different brands of machines. See your hardware documentation and your *XENIX Release Notes* for more information.

When you run **mkdev tape**, choose the option to "Install A Mini Cartridge Tape Driver" and then choose "y" when you are prompted to link the driver into your kernel.

Note that mini tape drives are not explicitly recognized with a message at boot time.

`/etc/default/tar`

After you install your mini tape drive, you must enter the correct size setting in the `/etc/default/tar` file. When you edit the file, you see several entries for various default devices. Here is the `/etc/default/tar` file provided with your distribution:

```
# device                block  size  tape
archive0=/dev/rfd048ds9  18     360   n
archive1=/dev/rfd148ds9  18     360   n
archive2=/dev/rfd096ds15 10    1200  n
archive3=/dev/rfd196ds15 10    1200  n
archive4=/dev/rfd096ds9  18     720   n
archive5=/dev/rfd196ds9  18     720   n
archive6=/dev/rfd0135ds18 18    1440  n
archive7=/dev/rfd1135ds18 18    1440  n
archive8=/dev/rct0       20     0     y
archive9=/dev/rctmini    20     0     y
# The default device...
archive=/dev/rfd096ds15  10    1200  n
```

Find the entry in your `/etc/default/tar` file for `/dev/rctmini`. In the above sample file, this is **archive7**. Note that the size value for **rctmini** is 0. You must change this entry when you install your **rctmini** device. The correct number for your **rctmini** device varies with the size of the tape you use. Here are the correct **size** settings for **rctmini** devices:

<u>Tape Size</u>	<u>Entry in Size field</u>
10 megabyte	8000
20 megabyte	18000
40 megabyte	39000

Note that you must include an even blocking factor (in this case "18") when you use **tar** on a mini tape drive. The utilities **backup** and **restore** have similar files and entries. For more information on default files, see **default(M)** in the *XENIX User's Reference* and the manual entry for the particular **backup** or **restore** command.

16.2.3 Using A Tape Drive

You use a QIC tape drive much like a floppy. You can use the standard commands such as **tar(C)**, **dd(C)**, **cpio(C)**, **backup(C)**, and **restore(C)**.

The tar Command

The **tar** command is useful for making a backup copy of entire directories. The command has the syntax:

```
tar cvf devicefile files
```

The *devicefile* is the file name that corresponds to the cartridge drive. *files* are the names of the files or directories to be copied. For example, to copy all the files in the directory */u/bogart* to the cartridge drive */dev/rct0*, enter:

```
tar cvf /dev/rct0 /u/bogart
```

Δ **sysadmsh** users select: Media→Archive

To restore files stored on tape, insert the cartridge containing the files or directories you wish to restore and enter the following command:

```
tar xvf devicefile
```

Δ **sysadmsh** users select: Media→Extract

tar restores all the files on the tape to the original directory.

For detailed information on backup operations and other methods of copying files, refer to **tar(C)**, **cpio(C)**, **backup(C)**, and **restore(C)** in the *XENIX User's Reference*.

Tape Drive Maintenance

The **tape(C)** utility performs various tape maintenance operations on standard QIC tape drives. **tape(C)** does not work with mini tape drives. **tape** sends commands and receives status from the tape drive. The basic form of the command is:

```
tape command [ devicefile ]
```


For example, to rewind the cartridge tape device `/dev/rct0`, enter:

```
tape rewind /dev/rct0
```

Other commands are:

- erase** Erase tape cartridge. Also re-tensions.
- reset** Reset tape controller and tape drive. Clears error conditions and returns tape subsystem to power up state.
- reten** Re-tension tape cartridge. Should be used periodically to remedy slack tape problems, generally resulting in an unusually large number of tape errors.

After certain tape operations are executed, the system returns a prompt before the tape controller has finished its operation. If you enter another tape command too quickly, the message “device busy” is displayed until the tape device is finished with its previous operation.

You should clean the tape drive heads and re-tension cartridges to keep it operating error-free.

Tape Formatting

Tape cartridges used with the mini tape drive (`ctmini`) must be formatted before use. Use the **format(C)** utility to format a cartridge tape. For example, this formats a `ctmini` tape cartridge:

```
format /dev/rctmini
```

△ **sysadmsh** users select: Media→Format

The **-e** option erases servo information on the 40 Megabyte cartridges:

```
format -e /dev/rctmini
```

Smaller cartridges must be bulk erased and reformatted. See also **tape(HW)** and **format(C)** for more information.



16.3 Using Floppy Disks

Floppy disks are the most convenient form of storage media. Depending on your floppy disk drive, you may be able to store from 360 kilobytes to 1.4 megabytes on a single disk. Floppy disks can be used for simple data storage in **tar**, **cpio**, **dd** or **dump** formats or you can make a mountable filesystem on a floppy disk. The following sections explain how to use floppies for data storage and as extra filesystem space.

16.3.1 Formatting Floppy Disks

Floppy disks must be formatted before they can be used. The XENIX command to format a floppy disk is:

```
format /dev/floppy-device
```

△ **sysadmsh** users select: Media→Format

The floppy device you specify in the command relates to the type of disk drive and floppy you are using. For example, if you have a high density 5.25 inch floppy disk drive, you can use it in high density mode (96 tracks per inch) or in low density mode (48 tpi). If you have high density floppies to use with your drive, the floppy device to specify is:

```
/dev/rfd096
```

16 In the above example, *rfd* indicates the raw floppy device, *0* indicates that this is the primary floppy drive, and *96* indicates high density mode. Similarly, if you wish to use low density floppies and the low density mode of the floppy drive, the device name is:

```
/dev/rfd048
```

In the above example, *48* indicates the low density mode of floppy drive *0*.

/etc/default/format file

You can also define a default format device by adding an entry to the file */etc/default/format*. For example:

```
DEVICE=/dev/rfd096ds15
```

After adding the above line, you no longer have to specify the device name. In addition, it is possible to define that all floppies be verified, which confirms that the data on the floppy is readable. (This can also be specified on the command line with the **-y** option.) Automatic verification can be specified by the following entry:

```
VERIFY=Y
```

If this entry is placed in */etc/default/format*, all floppies formatted with the **format** command are verified. (To override verification, use the **-n** on the command line.)

For more information, refer to **format(C)** in the *XENIX User's Reference*.

16.3.2 Copying Floppy Disks

To ensure against the loss of data stored on floppy disks, any user can use the **diskcp(C)** command, or the **dd(C)** command to make copies of floppy disks on new, formatted disks.

diskcp makes use of **dd** and provides a simple interface to that program. **dd** is very powerful, and you can use it to perform many different kinds of copying.

You must copy information onto formatted disks. If you format floppies under XENIX, you can use them over again without reformatting.

If you have disks that have been formatted under another operating system, you must reformat them under XENIX before you can use them to make copies of XENIX disks. Be aware that floppies formatted under some operating systems cannot be used under other operating systems, even with reformatting.



You can use the **format** command to format floppies. This command is described in the section “Formatting Floppy Disks” in this chapter. The **diskcp** can also format floppies for you. This is accomplished by the following steps.

To copy a floppy disk using **diskcp**, do the following:

△ **sysadmsh** users select: Media→Duplicate

1. Insert the disk you want to copy, also known as the *source* floppy, in drive 0, your primary floppy drive.
2. Insert another floppy in the other drive. This floppy is also known as the *target* disk. Note that any information already on the target disk will be destroyed.

If you have only one disk drive, leave the source floppy in the drive. **diskcp** will prompt you to remove the source disk at the correct time.

3. To format the floppy disk before the image is copied, enter the command:

diskcp -f

and press **RETURN**.

If your computer has dual floppy drives, enter the following command to copy the image directly on the target floppy:

diskcp -d

and press **RETURN**.

If you do not need to format the target floppy, simply enter:

diskcp

and press **RETURN**.

4. Follow the instructions as they appear on your screen. Note that, with a single drive system, you are prompted to remove the source disk and insert the target disk.



To copy a disk using **dd**, follow these steps:

1. Insert the disk to be copied into floppy drive 0.
2. Insert a formatted disk into drive 1. If necessary, you can format a disk with the **format** command described under “Formatting Floppy Disks” in this chapter.
3. Enter:

```
dd if=/dev/fd0 of=/dev/fd1 count=blkcount
```

and press **RETURN**. The *blkcount* is the number of blocks on the disk to be copied. If you do not know this number, leave the *count=blkcount* section out of the command.

This command copies the first disk to the second, then displays a record of the number of blocks copied.

16.3.3 Using Floppies for File Storage

To use a floppy for simple file storage, first, make sure that the floppy is formatted. Then, place the floppy in the floppy drive. You can use any of the standard XENIX file archiving utilities with floppy disks. These include **tar**, **cpio**, **dd** or **backup** formats.

tar is recommended for most file archiving tasks. For example, to place a copy of a file on a low density floppy disk in **tar** format, use the following command:

```
tar cv filename
```

Δ **sysadmsh** users select: Media→Archive

For more information on **tar**, see the **tar(C)** manual page in the *XENIX User's Reference*. For more information on **cpio**, **dd** and **backup** formats, see the associated manual pages in the *XENIX User's Reference*.



16.3.4 Making Filesystems on Floppy Disks

You can make a filesystem on a floppy disk much as you make one on a hard disk. Filesystems on floppy disks are portable and can be mounted on any XENIX system. XENIX provides a special directory called */mnt* especially for mounting filesystems that do not have a specified mounting point. Note that for system security, you must be logged in as "root" to use floppy filesystems.

In the example below, a filesystem is created on a high density 96tpi 5.25 inch floppy and mounted on the XENIX system.

To make a portable filesystem on a floppy disk, use the following procedure:

1. Enter the command:

```
mkdev fd
```

```
Δ sysadmsh users select: Filesystems→Add
```

You see the following menu:

```
Choices for type of floppy filesystem.
```

1. 48tpi, double sided, 9 sectors per track
2. 96tpi, double sided, 15 sectors per track
3. 135tpi, double sided, 9 sectors per track (3.5" diskette)

```
Enter an option or enter q to quit:
```

Enter **2** and press **RETURN**. You see the following prompt:

```
Insert a 96ds15 floppy into drive 0.
```

```
Press Return to continue or enter q to quit:
```

Press **RETURN**. Next you see:

```
Choices for contents of floppy filesystem.
```

1. Filesystem only
2. Bootable only
3. Root filesystem only
4. Root and Boot (only available for 96tpi floppy)

```
Enter an option or enter q to quit:
```

Enter **1** and press **RETURN**. Next you see:

```
Would you like to format the floppy first? (y/n)
```

If you have already formatted the floppy, enter **n** and the filesystem is immediately created. If the floppy has not yet been formatted, enter **y** and you see:

```
formatting /dev/rfd096ds15 ...  
track 00 head 0
```

The track and head numbers will count up as the floppy is formatted. When the formatting is finished, the word “done” is displayed. Next you see information about the filesystem as it is created. For a filesystem on a 96tpi floppy, you see:

```
isize = 288  
m/n = 1 15  
filesystem creation complete.
```

Next you see this menu again:

```
Choices for contents of floppy filesystem.

1. Filesystem only
2. Bootable only
3. Root filesystem only
4. Root and Boot (only available for 96tpi floppy)

Enter an option or enter q to quit:
```

Now enter **q** and press **RETURN** to quit. Your floppy now contains a filesystem. To use this filesystem, you must mount it on your system. To do this for a 96tpi floppy, give the following command:

```
mount /dev/fd096 /mnt
```

△ **sysadmsh** users select: Filesystems→Mount

Note that you use the floppy device *fd096* and not *rfd096*. When you mount a floppy filesystem, you must use the name without the preceding ‘r’. For another example, if you choose to mount a filesystem on a 48tpi disk, use the following command:

```
mount /dev/fd048 /mnt
```

When you give the **mount** command, XENIX should return a prompt. This indicates that the filesystem has been successfully mounted. You can now use the **cd** command to move into the filesystem and create files there normally. When you are done and you wish to remove the floppy, give the following command:

```
umount /dev/fd096
```

△ **sysadmsh** users select: Filesystems→Unmount

and your filesystem is immediately unmounted. Your files are contained on the floppy and can be stored or transported easily.

16.3.5 Creating an Emergency Boot Floppy

mkdev(ADM) provides a utility to create an Emergency Boot Floppy to allow you to restore a corrupted root filesystem without reinstalling XENIX. If you have more than one system, you should make one Emergency Boot Floppy for each machine. Because each machine has a unique "Emergency" floppy, a reboot floppy made on one system will not work with any other system. Be sure to keep these diskettes separate; if you use an emergency floppy on the wrong machine, it will not work and further corruption may result.

To create the floppy, **mkdev** uses a menu-driven program to select the disk format and filesystem type. There are three basic types generated: boot and root on a single disk (96 tpi only), boot and root pair (48 tpi), or filesystem only (as described above). The formats supported are: 48 tpi, 96 tpi-15 sectors/track, and 135 tpi-9 sectors/track in the 3 1/2 inch format. To create the floppies, enter:

```
mkdev fd
```

△ **sysadmsh** users select: Filesystems→Add

and you will see the following display:

```
Choices for type of floppy filesystem.
```

1. 48 tpi, double sided, 9 sectors per track
2. 96 tpi, double sided, 15 sectors per track
3. 135 tpi, double sided, 9 sectors per track (3.5" diskette)

```
Enter an option or enter q to quit:
```

Enter the correct type and press **RETURN**. You are then prompted for the disk:

```
Insert a x tpi floppy into drive 0. Press Return to continue.
```

XENIX System Administrator's Guide

Insert your floppy and press **RETURN**. The program responds with another menu:

```
Choices for contents of floppy filesystem.
```

1. Filesystem only
2. Bootable only
3. Root filesystem only
4. Root and Boot (only available for 96tpi floppy)

```
Enter an option or enter q to quit:
```

Select the appropriate filesystem and press **RETURN**. **fdinit** generates the filesystem and displays the following message when complete:

```
xx tpi filesystem floppy complete.
```



Chapter 17

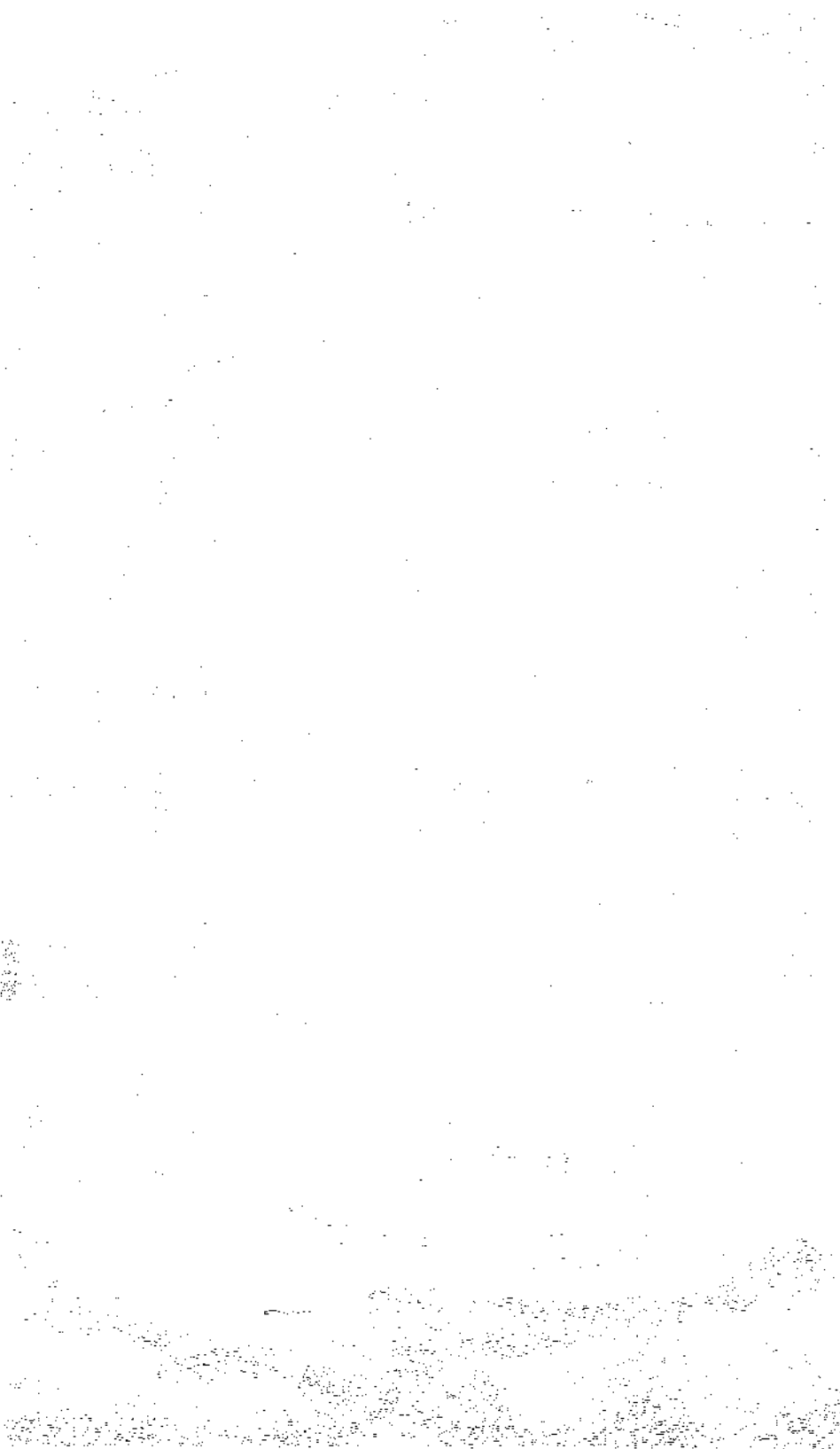
Using Bus Cards

17.1 Introduction 17-1

17.2 Installing Bus Cards 17-1

17.2.1 Using the Manufacturer's Setup Disk 17-2

17.3 Adding Additional Memory 17-3



17.1 Introduction

The bus (or “motherboard”) of your computer is the center of your system. Every system administrator must deal with the bus and the hardware associated with it. To find the bus on your system, you must generally remove the shell from the main body of your computer. Generally, you find a large circuit board with expansion slots for extra boards. These boards are commonly known as *Bus Cards*.

Bus cards can be extra memory for your system, internal modems, multi-port serial boards for extra terminals, controller boards for peripheral devices such as hard disks tape drives, control cards for monitors with color and graphics capabilities, mouse controllers, or other devices. In this chapter we explain a little about bus cards and how to install them with your XENIX system. Installation of most devices with bus cards is explained in detail in other chapters of this Guide, so we will deal only with the basics of bus cards in this chapter.

17.2 Installing Bus Cards

To install a bus card, you must first shut down XENIX and power down the system. Make sure that the computer is unplugged or you may injure yourself. Before you begin working on the computer, ground yourself by touching a metal object close at hand that is not the computer. Static electricity that builds up and jumps from your hand when you touch the hardware inside the computer can ruin your equipment.

Dip Switches and Jumpers

Before you plug your board into the bus, make sure that there are no settings on the board that must be changed. Again, your hardware documentation that comes with the board should list the default settings and how to change them. Generally, to change the settings of a board, there are dip switches and “jumpers.” Dip switches operate in “down” and “up” positions. Your hardware documentation should list the correct settings if your board has these switches. Jumpers are clips that slide over metal posts that stick out of the board to make a connection. You can change the settings on a board by moving the jumper to connect a different pair of posts. Again, your hardware documentation should provide you with specific instructions for jumper settings on your hardware.

Note

Note that XENIX is designed to work with most hardware using default settings. You will rarely have to change the settings on a board for it to work under XENIX.

Installing the Hardware

Carefully perform any steps necessary to expose the expansion slots on your computer. Your hardware documentation should explain this in detail. Once you can examine this area, note the number of available spaces for bus cards. A new system will have up to 8 or 10 available slots. Note that some slots are longer than others. There are both short and long cards. Short cards are about half as long as long cards. Generally there are 2 to 3 short slots and the rest are long slots. Find a slot that fits your board and gently but firmly plug the board into the slot in the bus. The board should have a tab on one side that fits into the slot on the bus. Bus cards only fit one way.

Some bus cards have a port that should face the outside of the computer. As stated before, bus cards only fit into the system one way. There may be a small plate covering an opening in the computer held on with a small screw. You can remove this cover plate if you need to. Boards such as modems, serial and parallel cards, and external device control cards will require this.

When you are done, replace the shell for your computer, and turn it on and boot. You may first need to use the manufacturer's setup program as described below to change the system's configuration before you can use the new hardware.

17

17.2.1 Using the Manufacturer's Setup Disk

Your computer should come with a manufacturer's setup program on a bootable floppy disk. Copy this disk for use and keep the original in a safe place. This disk is used to configure the permanent memory on your computer to describe the system hardware setup. Whenever you add a major device, like an extra hard disk or an extra serial card, you may need to run your setup program to tell your computer about the new hardware. Some

computers automatically recognize the presence of new hardware. Your manufacturer's documentation should let you know if you need to run this software.

17.3 Adding Additional Memory

You can improve system performance and run larger programs by increasing the amount of internal memory.

To increase internal memory follow these steps:

1. Turn off your computer.
2. Install extended memory according to the manufacturer's instructions. Make sure you have set all switches as noted in the instructions.
3. Boot XENIX. The boot screen details how the additional memory has affected your system.
4. Some features of XENIX may have been expanded. For example, you may have:
 - More multiscreens
 - More buffers
 - A larger maximum user process size

The number of multiscreens may be unchanged. Since the number of multiscreens can be set by the user, you may have already set a specific limit to the number of multiscreens available. If you have not set a limit to the number of multiscreens then you are already using the maximum number of multiscreens that XENIX allows.

17

The number of buffers may also be unchanged. Since the number of buffers can also be set by the user, you may have already set a specific limit to the number of buffers available. If you have not set a limit to the number of buffers then you are already using the maximum number of buffers that XENIX allows.

If the maximum user process size is unchanged, then it is now limited by the size of the *swap* filesystem instead of the amount of internal memory. You can:

- Reinstall XENIX and increase the size of the *swap* chapter of the *Installation Guide* for details on reinstallation.
- Change the process so that it runs without being swapped. Refer to **proct1 (S)** for details.

You can follow the same procedure if you wish to remove internal memory from the system.

If the memory hardware reports an error to XENIX, the following message is displayed:

```
PANIC: parity
```

You then see the software reboot message:

```
** Normal System Shutdown **  
  
** Safe to Power Off **  
    - or -  
** Press Any Key to Reboot **
```

If the system repeatedly panics from parity errors, consider replacing the memory chips.

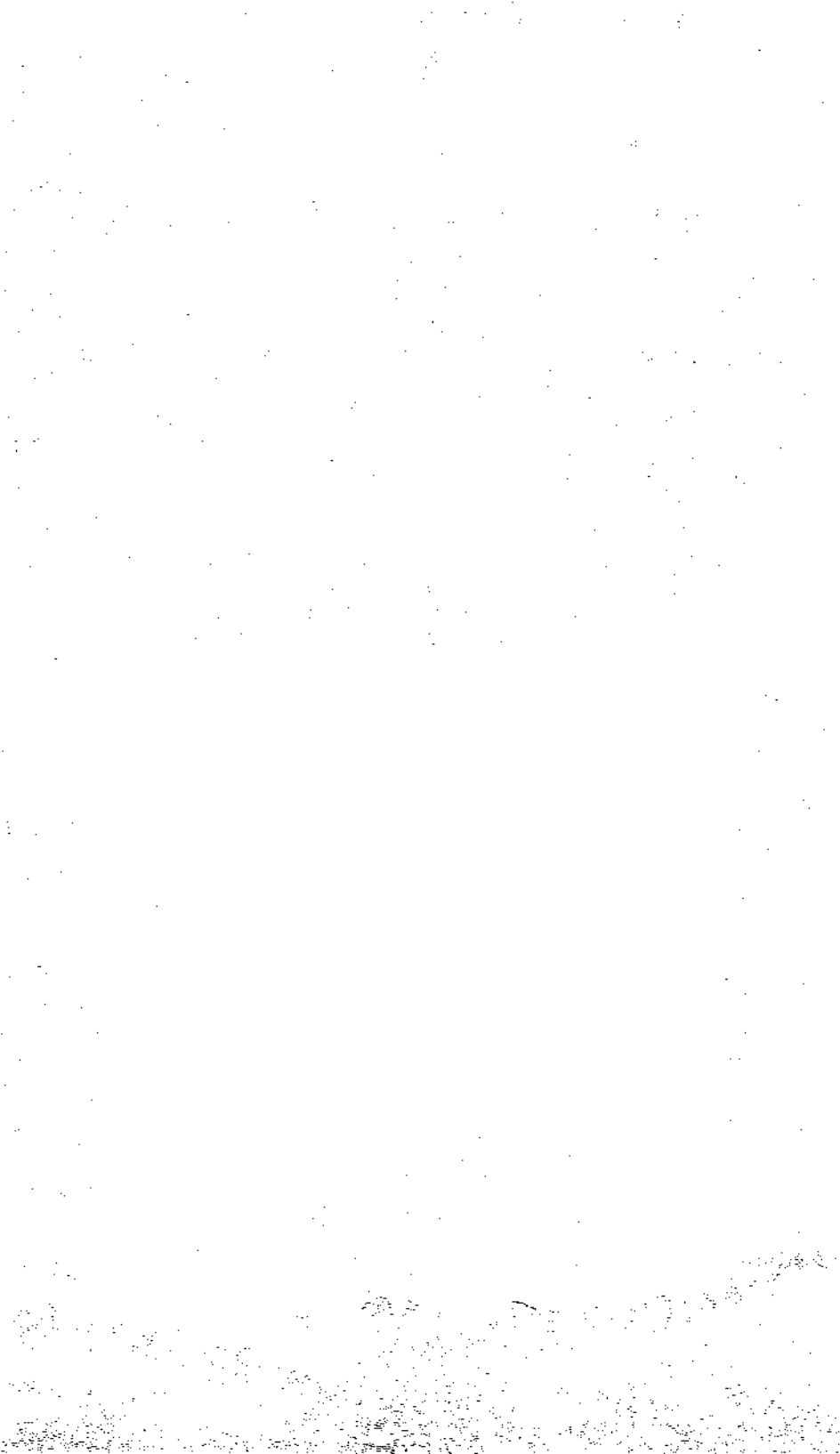
Note

Some machines have a hardware limitation on the maximum amount of memory that can be installed. Refer to your computer hardware manual to determine the maximum amount of memory you can install.

Chapter 18

Using a Mouse With XENIX

- 18.1 Introduction 18-1
- 18.2 Installing the Hardware 18-1
- 18.3 Installing A Mouse 18-1
 - 18.3.1 Removing a Mouse 18-6
- 18.4 Using the Mouse 18-6
 - 18.4.1 Using the Mouse With Multiscreens 18-6
 - 18.4.2 Using the Mouse on Serial Terminals 18-6
 - 18.4.3 Sharing a Mouse With Several Terminals 18-7
 - 18.4.4 Using a Mouse with Keyboard-based Programs 18-7



18.1 Introduction

This chapter deals with the basics of installing any brand or type of mouse interface with your system. Using a mouse can be a great convenience for users and developers alike. For this reason, XENIX provides support for both serial and bus mouse hardware.

18.2 Installing the Hardware

Consult your hardware manufacturer's documentation for specific instructions on hardware configuration. Note the brand and type of your mouse and whether it is attached to a serial port or directly to the system bus. For more information about the system bus, see the chapter in this Guide called "Using Bus Cards." You will need to know this information when you configure your software to accept the mouse.

18.3 Installing A Mouse

To install a mouse on your system, you must perform the following steps:

1. Install the mouse according to the manufacturer's instructions.
2. Make sure your link kit is installed and functioning correctly. The mouse drivers cannot be installed without the Link Kit.
3. Log in as root and input the following command:

```
mkdev mouse
```

And you see the Mouse Initialization Menu:

```
Mouse Initialization Program
```

1. Display current configuration
2. Add a mouse to the system
3. Remove a mouse from the system
4. Associate a terminal with an existing mouse
5. Disassociate a terminal from an existing mouse
6. Remove the mouse drivers from the kernel

```
Select an option or enter q to quit:
```

To install a mouse, choose option 2. The other options allow you to change your mouse configuration at any time. For example, you

can add or remove additional mice on your system or change the terminals that are allowed to receive input from an existing mouse.

4. Choose option 2 to install a mouse and press **RETURN**. Next, you must specify the type of mouse you will use. All the mice currently supported under XENIX are listed. Enter the number corresponding to your type of mouse and press **RETURN**. You see the menu:

```
The following mouse devices are supported:
```

1. Logitech serial mouse
2. Microsoft Serial Mouse
3. Mouse Systems PC Mouse
4. Microsoft Busmouse or INport Bus mouse
5. Logitech Bus Mouse
6. Olivetti Bus Mouse

```
Select an option or press 'q' to return to the previous menu:
```

Enter the number corresponding to the mouse you wish to install and press **RETURN**.

5. If you choose a busmouse, you are asked to select the configuration for the busmouse card. If you choose a serial mouse, you skip this step and go directly to linking the mouse drivers into your kernel. If you choose a busmouse, you see the message:

```
Configuring driver...
```

and then you see the menu:

```
Busmouse Configuration
```

1. Display current busmouse parameters
2. Modify current busmouse parameters
3. Select previous busmouse parameters
4. Select default busmouse parameters

```
Enter an option or q to quit:
```

If you wish to use the default busmouse parameters, select option 4 and then press “q” to quit this menu. The default busmouse selection allows XENIX to auto-configure your busmouse. If you choose alternate parameters, be sure that you use interrupt vector 5 with your busmouse. Note that using interrupt vector 5 may conflict with a cartridge tape device if both devices are in use at the same time.

6. If you have previously installed a mouse of any sort on your system, the driver for the mouse devices should already be linked with your kernel. If you have never installed a mouse on your system, or the driver is not present in the kernel, you see the following messages. Note that these messages may take a few minutes to appear on your screen:

```
Updating system configuration...
Installing mouse drivers...
You must create a new kernel to effect the driver change
you specified.
Do you wish to create a new kernel now? (y/n/q)
```

Answer “y” to add the mouse device driver to your kernel. As part of the linking process, you see the following messages:

```
Kernel with mouse driver modification is in
/usr/sys/conf/xenix
Do you want this kernel to boot by default? (y/n)
```

Answer “y” if you want this kernel to be used every time you boot XENIX. (Most systems will answer “y” to this prompt.) Next you see:

```
The new kernel is installed in /xenix.
After you finish configuring the mouse,
reboot your system to activate the new kernel.
```



You have now installed the mouse drivers in your kernel.

7. Next, you are asked to specify the terminals and multiscreens that will be allowed to accept input from the mouse. Do not attempt to allow mouse input on any tty where any mice are physically connected or you will receive an error message. You may choose to

allow any or all other terminals and video adapter monitor multiscreens to use the mouse. Entering the word "multiscreen" will associate all of the console multiscreens.

Note that only one mouse can be allowed for input on a given tty.

For more information on sharing the mouse between several terminals or multiscreens, see "Using the Mouse." You see:

```
Enter a list of terminals (e.g. tty1a, tty2a, multiscreen)
or enter 'q' to quit. Press return when finished:
```

When you have pressed return, you see:

```
Do you want to use the <mouse_type> on any other terminals?
(y/n/q)
```

Note that in above example, *mouse_type* will be replaced with the brand or type of mouse you specified earlier in the procedure.

8. Next you see:

```
Do you want to use the <mouse_type> on any other terminals?
(y/n/q)
```

If you answer "n" no other terminals will be allowed to receive mouse input. If you answer "y" you are prompted to enter tty pathnames for terminals that are to be allowed to receive mouse input. You see:

```
Enter a list of terminals (e.g. tty1a tty1b). Press return
when finished:
```

9. Next, if you are installing a serial mouse, you are asked to specify the tty port where you will physically attach the mouse. If you are installing a bus mouse, proceed to the following step, as the port is configured automatically. You see:

```
<mouse-type> is currently configured to attach to the
system on /dev/tty1a.
```

```
Do you wish to install this mouse on a different port? (y/n/q)
```

Note that you can choose any port on your system for your mouse, but that the default port is /dev/tty1a (COM 1). Otherwise, enter the serial tty name (for example, /dev/tty5a or /dev/tty2b) where you will attach your mouse. You see:

```
To which device (e.g. /dev/tty2a) do you want to attach the
<mouse_type> (or enter 'q' to quit):
```

10. Finally, you are returned to the main mouse menu again:

```
1. Display current configuration
2. Add a mouse to your system
3. Remove a mouse from your system
4. Associate a terminal with an existing mouse
5. Disassociate a terminal from an existing mouse
```

```
Select an option or enter 'q' to return to the previous menu:
```

If you have no changes to make to your mouse configuration at this time, enter “q” to quit and press **RETURN**.

Note that you can invoke **mkdev mouse** at any time to allow or prevent input on different terminals, remove mice or check your current configuration.

18.3.1 Removing a Mouse

Removal of any mouse or the mouse drivers on your system is an exact reversal of the process of installing a mouse. Choose the menu options to remove rather than to add a mouse.

18.4 Using the Mouse

Using the mouse under XENIX is automatic. If a program or utility accepts mouse input and the terminal is allowed to use the mouse, you simply invoke the program and the mouse works. If the terminal or multiscreen is not allowed to use the mouse, or the program is not configured to accept mouse input, using the mouse has no effect.

18.4.1 Using the Mouse With Multiscreens

Multiscreens (on monitors attached to video cards in the bus) provide the most convenient method for using the mouse. If a mouse is associated with the multiscreens on your main system console, (typically, a monitor attached to a video card in the system bus) the mouse input is associated with the current active multiscreen. For example, if your system has four multiscreens enabled on the main system console and all those screens are allowed to use the mouse, the input from the mouse goes to the program running on the active multiscreen.

Remember that programs that do not accept mouse input will be unaffected by moving the mouse, even on a mouse-allowed multiscreen.

Serial (or terminal) multiscreens and serial consoles can also be configured to use the mouse.

18.4.2 Using the Mouse on Serial Terminals

When you install the mouse, you are prompted to list the ttys that will be allowed to use mouse input. You can allow terminals on serial lines to use the mouse just as you allow multiscreens. Again, note that you cannot allow mouse input on a tty where a mouse is physically connected.



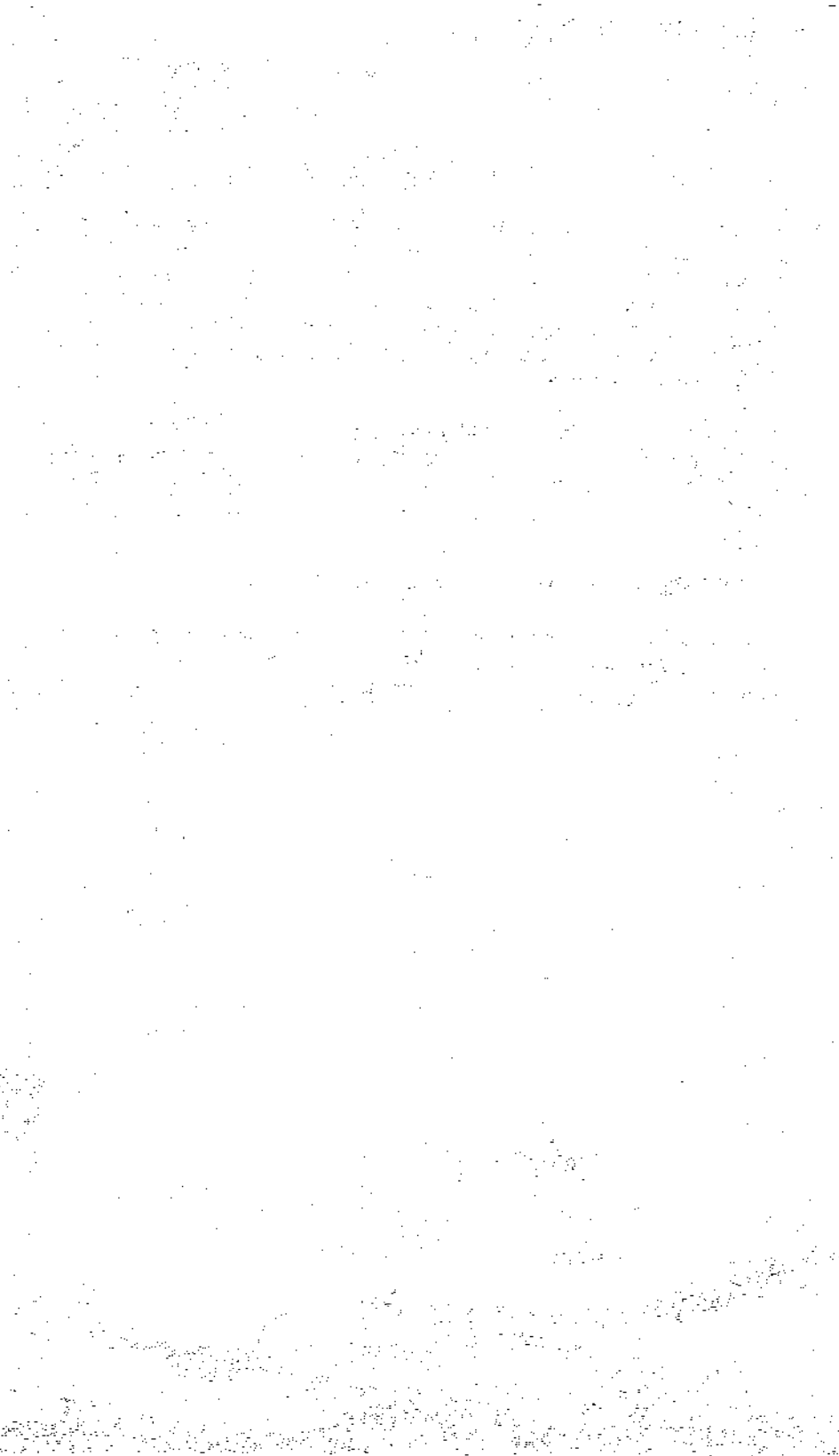
18.4.3 Sharing a Mouse With Several Terminals

When the mouse is shared among several terminals, the mouse is associated with a tty on a “first-come, first-served” basis. The first user to invoke a mouse enabled program has the mouse for the duration of that program. In order for another user to use the mouse, the first user must quit the program. (Thereby closing the input queue from the mouse.) Then, the next user for the mouse can invoke the program and open the line for input from the mouse.

Note that other users on mouse-allowed ttys can use programs that accept mouse input while the mouse is busy. If the mouse is busy, the programs will be unable to receive input from the mouse but should otherwise function normally.

18.4.4 Using a Mouse with Keyboard-based Programs

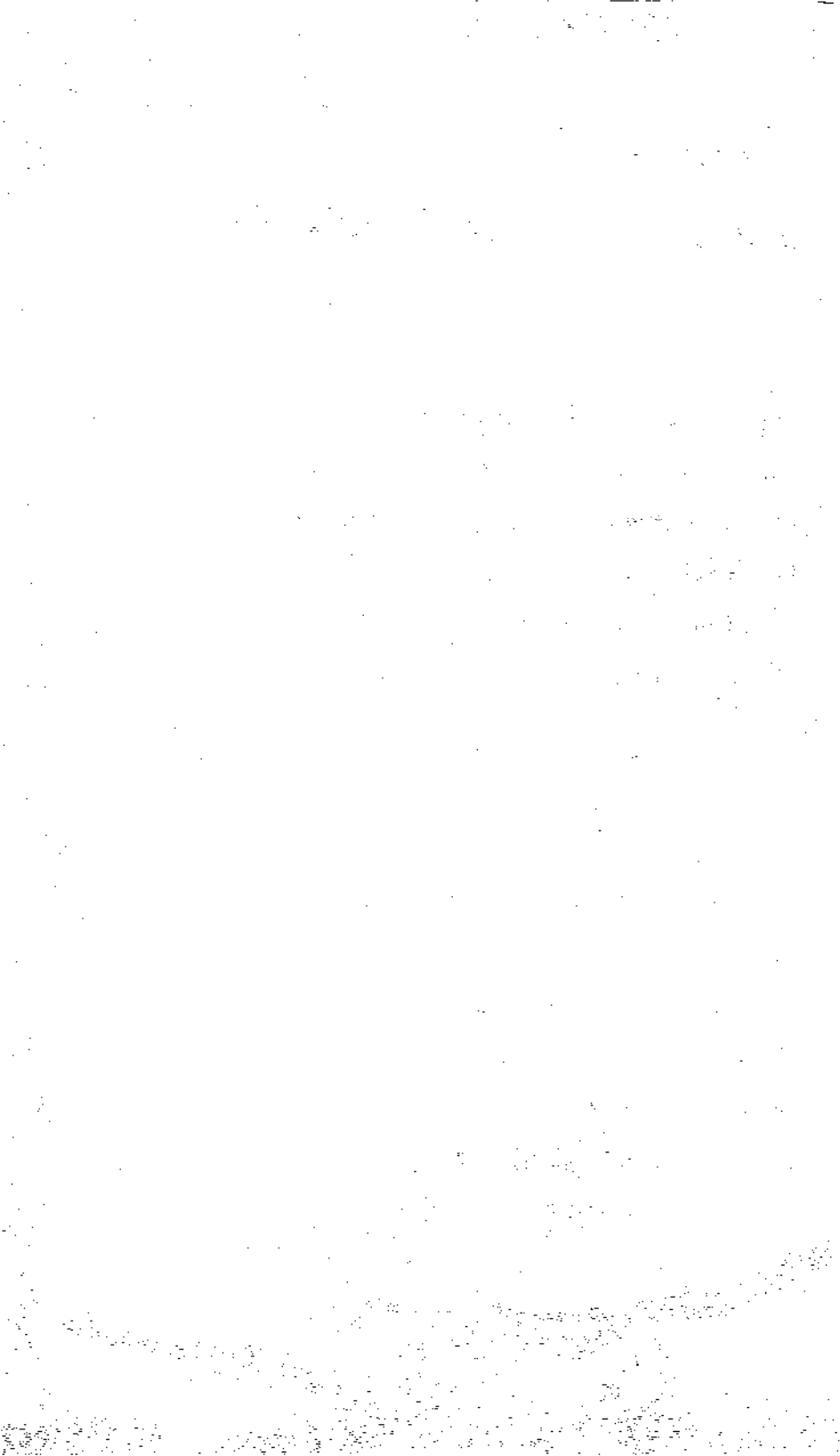
The **usemouse(C)** utility is used to map mouse movements and operations to keystrokes used by keyboard-based programs. Refer to the **usemouse(C)** manual page for complete information.



Chapter 19

Solving System Problems

- 19.1 Introduction 19-1
- 19.2 Restoring a Nonechoing Terminal 19-1
- 19.3 Restarting a Locked Terminal 19-1
- 19.4 Loading adb(CP) Without the Development System 19-4
- 19.5 Fixing Console Keyboard Lock-up 19-6
- 19.6 Restarting a Stopped Printer Queue 19-8
- 19.7 Fixing a Slow Parallel Printer 19-9
 - 19.7.1 Switching to Polling Operation 19-10
- 19.8 Stopping a Runaway Process 19-11
- 19.9 Replacing a Forgotten User Password 19-12
- 19.10 Removing Hidden Files 19-12
- 19.11 Restoring Free Space 19-13
- 19.12 Restoring Lost System Files 19-13
- 19.13 Restoring a Corrupted root Filesystem 19-13
- 19.14 Recovering from a System Crash 19-14
- 19.15 Fixing Bad HZ Value 19-15
- 19.16 Recovering from a General Protection Trap 19-16
- 19.17 Mapping a Bad Track 19-17



19.1 Introduction

This chapter explains how to solve problems that affect the operation of the system. The tasks range in complexity from how to fix a nonechoing terminal, to restoring lost system files.

19.2 Restoring a Nonechoing Terminal

A nonechoing terminal is a terminal that responds to keyboard input but does not display the display characters entered at the keyboard. (This is opposed to a “locked” terminal that does not respond to input at all; see “Restarting a Locked Terminal” for a solution to this problem.) This abnormal operation can occur whenever a program stops prematurely as a result of an error, or when the user presses the BREAK key.

To restore the terminal to normal operation, follow these steps:

1. Press **Ctrl-j**. The system may display an error message. If it does, ignore the message.
2. Enter:

stty sane

and press **Ctrl-j**. The terminal does not display what you enter, so be sure to enter it accurately.

After pressing Ctrl-j, the terminal should be restored and you may continue your work.

19.3 Restarting a Locked Terminal

One of the most frustrating problems faced by new users is a locked terminal. Terminals may lock for a variety of reasons, and the results range from momentary interruptions to lost work. When the problem occurs, the following steps should return the terminal to use with a minimum of lost time.

1. The first step is to wait. As your computer system becomes more heavily used, the “response time” will increase. During periods of peak activity, the terminal may accept keystrokes without “obeying” them, giving the impression that it has locked completely. Wait a minimum of 60 seconds before trying to resurrect the terminal.

2. Press **Ctrl-q** to re-enable transmission, in case a **Ctrl-q** (transmit off) signal, such as from a "No Scroll" key, has inadvertently been sent to the computer.
3. If the terminal remains unresponsive, it should be checked for hardware problems, such as a loose or disconnected power cord, keyboard cord, or communications cable. If everything is plugged in, and the communications cable is tight at both ends (and still intact in the middle), look for a software problem.
4. Sometimes the terminal's internal software may have become stuck in an unusual state. This can often be corrected by turning the terminal off and back on again. This will *always* result in a blank screen, so if there is information on the screen that you wish to save, you may want to copy it by hand, or save this step for last.
5. Having eliminated the wiring and the terminal itself, as sources of the problem, check the programs running on the terminal. The **who** command, typed on an operating terminal, will tell you which communication port each user is connected to, allowing you to identify the port of the locked terminal.

A useful hardware-communications test is redirecting some output from an operating terminal to the locked one. For a communication port named **tty6c**, a command such as:

```
date > /dev/tty6c
```

should produce some output on the screen of the locked terminal. If you get the message "**permission denied**," have the user log in on the operating terminal, and try the **date** command again. If there is still no output, go back and check the hardware again as described above.

To find out what programs the locked terminal is running, give the name of the port to the **ps** command, using the **-t** option. For the communication port **tty6c**, as above, the command:

```
ps -t tty6c
```

will list the programs being run by the terminal on that port. (Again, this must be typed on an operating terminal.) Also listed will be each program's PID (process identifier). Make a note of each program name and PID number.

Determining which program is currently running on the terminal may sometimes take a little guesswork. Often it will be the last

line of the **ps** list or the program with the highest PID. Other clues may be obtained by using the **-f** and **-l** options to **ps** (combined with **-t** as **-flt**) which will list the starting times (**STIME**) and the states (**S**), respectively, of the processes. (For more information see **ps(C)** in the *XENIX User's Reference*.)

You may find that the user is running a different program than they think. This can happen, for instance, when a user accidentally runs a program from a menu. In this case, the locked terminal may be freed by using the proper commands to exit the unwanted program.

If the currently-running program is the proper one, and the terminal will not respond to correct commands for that program, then the program must be "killed." Only the user who started the program or the super user (**root**) may kill a user's programs, so have the user log in on the operating terminal, if they have not already done so (or log in as **root**).

The **kill** command normally takes two arguments, a *signal* and a PID (see **kill(C)** for more details). For a currently-running program with a PID of 1234, the command

```
kill -15 1234
```

will send process **1234** a signal of **15**, which asks the program to leave "politely," and will cause a minimum of problems (if it works).

Now use the **ps -t** command again to see if the program you "killed" has exited. If it has not stopped, issue the **kill** command again using **-9** instead of **-15**. This is a "sure kill," and may cause remnants of the program to be left behind, such as temporary files. You should check for these in the **/tmp** and **/usr/tmp** directories when you have finished, looking for files owned by the user, and remove them. Note that files belonging to programs which have not been killed may also be in these directories. If in doubt, leave it there and clean up later.

After each program is killed, the terminal may be tested. It may not respond. Many programs place the terminal in special modes when they run, and some of these modes will cause the terminal to seem to be locked. Applications may use what is called "raw" mode, in which characters typed at the terminal are not echoed to the screen. One problem with raw mode is that the RETURN or ENTER keys do not behave as expected.

XENIX expects a **Ctrl-j** (linefeed or newline) character at the end of each command, but most terminals send a **Ctrl-m** character when the RETURN key is pressed. Normally, the **Ctrl-m** is translated into **Ctrl-j**, but if the application has turned off this translation, and has then exited or been killed without turning the translation back on, you must type a true **Ctrl-j** character to end your commands.

On the locked terminal, type a **Ctrl-j** to start a new line, and then enter:

```
stty sane <Ctrl-j>
```

Be sure to use **Ctrl-j** instead of the RETURN or ENTER key to end the command. You may need to type this command twice before the terminal responds.

The kill-and-test sequence above may need to be used on *each* program listed on the the output of the **ps** command. Many system administrators will simply kill *all* of the programs to save time. This choice is yours.

If the **ps -t** command shows only a program named **getty**, you have killed all the possible programs, and should have a login prompt on the terminal. If not, go back and check the hardware.

Note

Some programs cannot be killed. This is rare and there is only one solution: the computer must be shut down and restarted.

19.4 Loading adb(CP) Without the Development System

adb is provided as part of the Operating System distribution in order to apply kernel patches of the type described in "Fixing Console Keyboard Lockup." Apply patches only when they are recommended by the *Release Notes* or other documentation relating to your distribution. In addition, you should keep a record of all patches applied in your system log book.

To load **adb** on your system, enter the following command to determine which N volume of your XENIX distribution it is on:

```
grep adb /etc/perms/dsmd
```

This command displays the volume number in the last column; for example:

```
SOFT x711 bin/bin 1 ./bin/adb N01
```

In this case, **adb** is found on volume N1. Use the following key to determine whether the N volume must be mounted and **adb** copied off, or extracted from the volume using **tar**:

Distribution	Type of Volumes
48/135tpi	N1 and N2 mountable; all others are tar volumes
96tpi	N1 is mountable; all others tar

6. Insert the proper N volume into your drive. Use one of the following commands to retrieve **adb** from your distribution. If the N volume is mountable, copy **adb** using the following commands:

```
mount -r /dev/install /mnt  
cp /mnt/bin/adb /bin  
umount /dev/install
```

If the N volume is a **tar** volume, enter the following commands:

```
cd /  
tar xvf /dev/install ./bin/adb
```

Note

adb is not supported for general programming and debugging unless you have purchased the Development System.

19.5 Fixing Console Keyboard Lock-up

A very small number of systems experience a problem known as “keyboard lockup,” where the system does not respond to keyboard input from the console keyboard. This has been investigated thoroughly and should be quite rare. This particular condition only affects keyboards that are attached to the video display adapter, not standard terminals that are attached to serial lines.

Your keyboard may be “locked up” if:

- The system console keyboard cannot be used to enter data or perform any tasks
- You cannot flip Multiscreens and the CAPS LOCK key does not turn the caps lock light on or off
- Other terminals on the system continue to work
- Printers or other devices continue to work
- The system is still running.

Keyboard lockup is similar to other circumstances, so before trying to fix a “locked” keyboard, make sure that:

- You did not accidentally enter Ctrl-S
- The “Keyboard Lock” key is not in the lock position
- The keyboard is still plugged in
- The system itself is still running.

First, make sure you did not enter Ctrl-S accidentally. Press Ctrl-Q several times and check to see if you can enter characters on the screen. Press RETURN a few times, or enter DEL.

Next, check the Keyboard Lock key, if your computer has one. It should be turned to the “unlocked” position. Also, make sure the keyboard is still plugged in to the correct socket.

19

Make sure the system is still running. Check a terminal to see if it is still working and that you can perform system tasks, such as logging in and checking the date. If you do not have a terminal, watch the hard disk access light, if your computer has one. If it flashes periodically, at least once every thirty seconds or so, the system is still running and is using the

hard disk. Note that you cannot use other terminals and that the hard disk access light may not flash if you are in single user mode.

If you check all of the suggestions, but you still cannot use your console keyboard, try unplugging the console keyboard then plugging it in again. If this fixes the problem, it is definitely a case of keyboard lockup. If this last step does not fix the problem, you may still have keyboard lockup.

You can prevent keyboard lockup by applying a special “patch” that changes the operating system kernel. (The kernel is the main program of the operating system that is always running in memory.) Note that this patch disables the keyboard lights (LEDs), so you should do this only if you have tried all other approaches:

1. Get the system console working, if it is not. Reboot the system if you have to and bring it up to single-user mode.
2. If you didn't reboot, log in as root on the system console and shut the system down to single user mode with the **shutdown** command:

```
/etc/shutdown su
```

See **shutdown(ADM)** for more information.

3. Once the system is in single user mode, back up the kernel:

```
cd /  
mv xenix xenix.00  
cp xenix.00 xenix
```

4. If you do not have the Development System installed on your system, load the **adb** program on your system as described in the section “Loading adb(CP) Without the Development System.”
5. When you have **adb** installed, enter this command:

```
adb -w /xenix  
ledspresent/w 0  
$q
```

This patches your kernel with the necessary fix. Again, note that it permanently disables the console keyboard lights (LEDs).

6. Shut down the system:

```
# /etc/shutdown 0
```

7. When you see the "Normal System Shutdown" message, press any key to reboot the system. You have now fixed the keyboard lockup problem.
8. Finally, call your support center and report your problem.

19.6 Restarting a Stopped Printer Queue

No printing can be done on the lineprinter spooling system unless the print scheduler, **lpsched**, is running. To check the status of **lpsched**, enter:

```
lpstat -r
```

△ **sysadmsh** users select: System→Report→Printers

To restart **lpsched**, enter the following on two lines:

```
/usr/lib/lpshut  
/usr/lib/lpsched
```

Calling the file */usr/lib/lpshut* cleans the system and */usr/lib/lpsched* starts it up again.

Access to files and directories in */usr/spool/lp* by **lp** can be another source of spooling problems. You can check the **lpsched** log file, */usr/spool/lp/log*. This is a record of the print scheduler's activity and errors. If **lpsched** refuses to run or a printer refuses to print, check to make sure that:

- The printer is enabled; see **lp(C)** in the *XENIX User's Reference*.
- The files and directories in */usr/spool/lp* are readable and writable by **lp**.

For more information on the lineprinter spooling system, see the section "Installing A Lineprinter" in the "Using Printers" chapter.

19.7 Fixing a Slow Parallel Printer

If you have a parallel printer that prints abnormally slow, check your configuration according to the procedure described below. If it still prints slowly, you can switch to polling operation. Verify the following items, as it is important that your parallel ports be configured properly to work under XENIX:

1. The printer must be IBM compatible and should use a standard Centronics interface cable in order for it to work at all.
2. The IBM AT and compatibles only support up to 2 parallel ports. Deconfigure additional ports.
3. The parallel port on the monochrome card should be configured for interrupt vector 7, and is recognized as **lp1** when booting up.
4. The main parallel port should be configured for interrupt vector 7 and is recognized as **lp0**. So you must use either the main or the monochrome's port - not both - to avoid a hardware conflict which would cause slow printing.
5. The alternate/second parallel port should be configured for interrupt vector 5, and recognized as **lp2**. Make sure no other hardware is using these interrupts. (See your hardware manual for information on configuring your parallel ports.)
6. Check to make sure your parallel card is recognized by XENIX by rebooting your system. Following the copyright information on the screen, you should see:

```
Parallel port lpn present
```

where **lpn** is the device special filename of the port the printer is attached to (**lp0**, **lp1** or **lp2**) as described above. If you do not see this message, check the switches and jumpers on your parallel card to make sure they are correct. Also try setting the card for a different configuration, if possible.

7. Be sure your printer is turned on and on-line, and the cable is properly attached to the computer and printer.
8. From the console, logged in as root, see if you can redirect output to your printer by entering:

```
date > /dev/lpn
```

If you do not see the date printed on your printer, there is most likely some type of hardware malfunction, so verify the following:

- Your cable is securely connected and all wires are good. Using the cable on a known good system, or printing under DOS are good ways to test this.
- Re-check your printer configuration by verifying its switches in your printer hardware manual.
- Re-check the switches on your parallel card. It must be recognized at bootup, as described above in Step 6.

When you get the date printed on your printer, you should run **mkdev lp** to configure the spooler to support this printer. (See the "Using Printers" chapter of this guide for more information on **lpinit**, the script called by **mkdev lp**.)

19.7.1 Switching to Polling Operation

If your parallel ports are configured properly, as described in the previous section and you still get slow printing, your parallel port may not be capable of generating interrupts.

A solution is to alter the way that the hardware and the printer driver communicate. The parallel printer driver can be made to "poll" a parallel port. This way the driver does not rely on interrupts from the parallel port, but at the cost of a possible drain on system resources.

To set up polling for a parallel port/parallel printer, create what is known as a "special device node." Log in as **root** (super user) and enter one of the following sets of commands. (Note which printer ports are recognized during the boot up message.)

For lp0:

```
mknod /dev/lp0p c 6 64  
chown bin /dev/lp0p  
chgrp bin /dev/lp0p  
chmod 222 /dev/lp0p
```

For lp1:

```
mknod /dev/lp1p c 6 65
chown bin /dev/lp1p
chgrp bin /dev/lp1p
chmod 222 /dev/lp1p
```

For lp2:

```
mknod /dev/lp2p c 6 66
chown bin /dev/lp2p
chgrp bin /dev/lp2p
chmod 222 /dev/lp2p
```

If you are using the print spooler, run **mkdev lp** to inform the spooler of the new parallel poll device. You can choose to add a new printer or re-configure an existing printer. When you are asked to choose a device for the printer, do not use the standard parallel devices that are displayed. Instead, use: “/dev/lp0p”, “/dev/lp1p”, or “/dev/lp2p”.

19.8 Stopping a Runaway Process

A runaway process is a program that cannot be stopped from the terminal at which it was invoked. This occurs whenever an error in the program “locks up” the terminal, that is, prevents anything you enter from reaching the system.

To stop a runaway process, follow these steps:

△ **sysadmsh** users select: Processes→Terminate

1. Go to a terminal that is not locked up.
2. Log in as the super user.
3. Enter:

```
ps -a
```

and press the **RETURN** key. The system displays all current processes and their process identification numbers (PIDs). Find the PID of the runaway program.

4. Enter:

```
kill PID
```

and press the **RETURN** key. The *PID* is the process identification number of the runaway program. The program should stop in a few seconds. If the process does not stop, enter:

```
kill -9 PID
```

and press the **RETURN** key.

The last step is sure to stop the process, but may leave temporary files or a nonechoing terminal. To restore the terminal to normal operation, follow the instructions in the section "Restoring a Nonechoing Terminal" in this chapter.

19.9 Replacing a Forgotten User Password

The XENIX operating system does not provide a way to decipher an existing password. If a user forgets his password, the system manager must change the password to a new one. To change an ordinary user password, follow the instructions in the section "Changing a User's Password" in the "Maintaining System Security" chapter.

19.10 Removing Hidden Files

A hidden file is any file whose name begins with a dot (.). You can list the hidden files in a directory by entering:

```
lc -a
```

and pressing the **RETURN** key.

You can remove most hidden files from a directory by entering:

```
rm .[a-z]*
```

and pressing the **RETURN** key. Remaining files can be removed individually.

19.11 Restoring Free Space

The system displays an “out of space” message whenever the root directory has little or no space left to work. To restore system operation, you must delete or reduce one or more files from the root directory. To delete and reduce files, follow the steps outlined in the section “Maintaining Free Space” in the “Using Filesystems” chapter of this guide.

19.12 Restoring Lost System Files

If a system program or data file is accidentally modified or removed from the filesystem, you can recover the file from the periodic backup disk with the `sysadmin` program. To restore the files, follow the instructions in the section “Restoring Individual Files or Directories from Backups” in the “Backing Up Filesystems” chapter of this guide.

19.13 Restoring a Corrupted root Filesystem

On very rare occasions, one or more of the critical XENIX system files may be accidentally modified or removed, preventing the system from operating. In such a case, you must restore your root filesystem from backups. To restore your root filesystem, you must first have made an Emergency Boot Floppy as you were directed in the *XENIX Installation Guide*. If you have not made this floppy, you must reinstall XENIX. Also, if you have no backups of your root filesystem, you must reinstall XENIX. To reinstall the system, follow the instructions in “Reinstalling and Updating Your System” in the *XENIX Installation Guide*.

To restore your root filesystem, perform the following steps exactly:

1. Power cycle your machine and boot the system using your Emergency Boot Floppy for that machine. (Note that you must have a separate Emergency Boot Floppy for each XENIX system or further corruption may result.)
2. At your system prompt, give the command:

```
/bin/fsck -y /dev/hd0root
```

You should see messages indicating that **fsck** is proceeding through five or six phases of system cleaning. If the program exits within a few seconds or you see error messages that make no sense, such as:

```
UNKNOWN FILE SYSTEM VERSION 65535
CLEANING NON SYSTEM 3 FILESYSTEM
```

you must restore your entire root filesystem. If **fsck** appears to be successful, shut down your system using */etc/haltsys* and attempt to boot from the hard disk. If this is not successful, you must continue this procedure.

3. If the above procedure does not correct your problem, you must restore your root filesystem. At your system prompt, enter the following command:

```
/etc/mount /dev/hd0root /mnt
```

4. Next, give the command,

```
restor fr /dev/devicename /dev/hd0root
```

Where *devicename* is the name of the device where you will read your backups. For example, a cartridge tape drive would be known as */dev/rct0* and a mini-tape drive would be */dev/rctmini*. A 96tpi floppy drive would be known as */dev/fd096*.

5. After the restoration is complete, halt the system using */etc/haltsys* and reboot from the hard disk. You should now be able to restore other filesystems normally. If you cannot boot from your hard disk at this point, it is likely that you have serious hardware malfunctions.

19.14 Recovering from a System Crash

A system crash is a sudden and dramatic disruption of system operation that stops all work on the computer. System crashes occur very rarely. They are usually the result of hardware errors or damage to the root filesystem which the operating system cannot correct by itself. When a system crash occurs, the system usually displays a message explaining the cause of the error, then stops. This gives the system manager the chance to recover from the crash by correcting the error (if possible), and restarting the system.

A system crash has occurred if the system displays a message beginning with “panic:” on the system console, or the system refuses to process all input (including INTERRUPT and QUIT keys) from the system console and all other terminals.

To recover from a system crash, follow these steps:

1. Use the error message(s) displayed on the system console to determine the error that caused the crash. If there is no message, skip to step 3.
2. Correct the error, if possible. A complete list of error messages and descriptions for correcting the errors is given in **messages(M)** in the *XENIX User's Reference*. (Even if the problem cannot be located or corrected, it is generally worthwhile to try to restart the system at least once by completing the remaining steps in this procedure.)
3. Turn off the computer and follow the steps described in the “Starting the System” chapter, to restart the system.
4. If the system will not restart, or crashes each time it is started, the operating system is corrupted and must be restored or reinstalled. Follow the procedures described in the previous section to restore the system and in the “Backing Up Filesystems” chapter, to restore user files.
5. If the system cannot be started from the “Boot” disk in the distribution set for installation, the computer has a serious hardware malfunction. Contact a hardware service representative for help.

19.15 Fixing Bad HZ Value

If you see the message “Bad HZ Value” at any time during system operation, your system files may be incorrect or corrupted.

The HZ variable is used by XENIX to represent the system interrupt clock frequency. You must declare the HZ variable in three places; in the */etc/rc* file, the */.profile* or */.login* file, and the */etc/default/login* file. If the HZ variable is not correctly set in these files or one or more of these files is corrupted or missing, you will see the above error message. Also, if you verify that these files are correct, this message could indicate that your kernel is not properly serialized.

If you have an 80286 or 80386 system, the HZ value must be set to 50 cycles per second. If you have a 8086 system, the HZ value is set to 20 cycles per second.

If you verify that the value is set correctly in your system files, you can re-serialize your kernel. To re-serialize your kernel, enter the following commands:

```
cd /  
cp /xenix xenix.bkp  
/etc/brand <serial#> <activationkey> /xenix  
/etc/shutdown
```

Now reboot your system. You should not see the error message.

19.16 Recovering from a General Protection Trap

Under XENIX the 80286 processor runs in "Protected" mode. If you have an 80286 based system and you see the error message:

```
panic: general protection trap
```

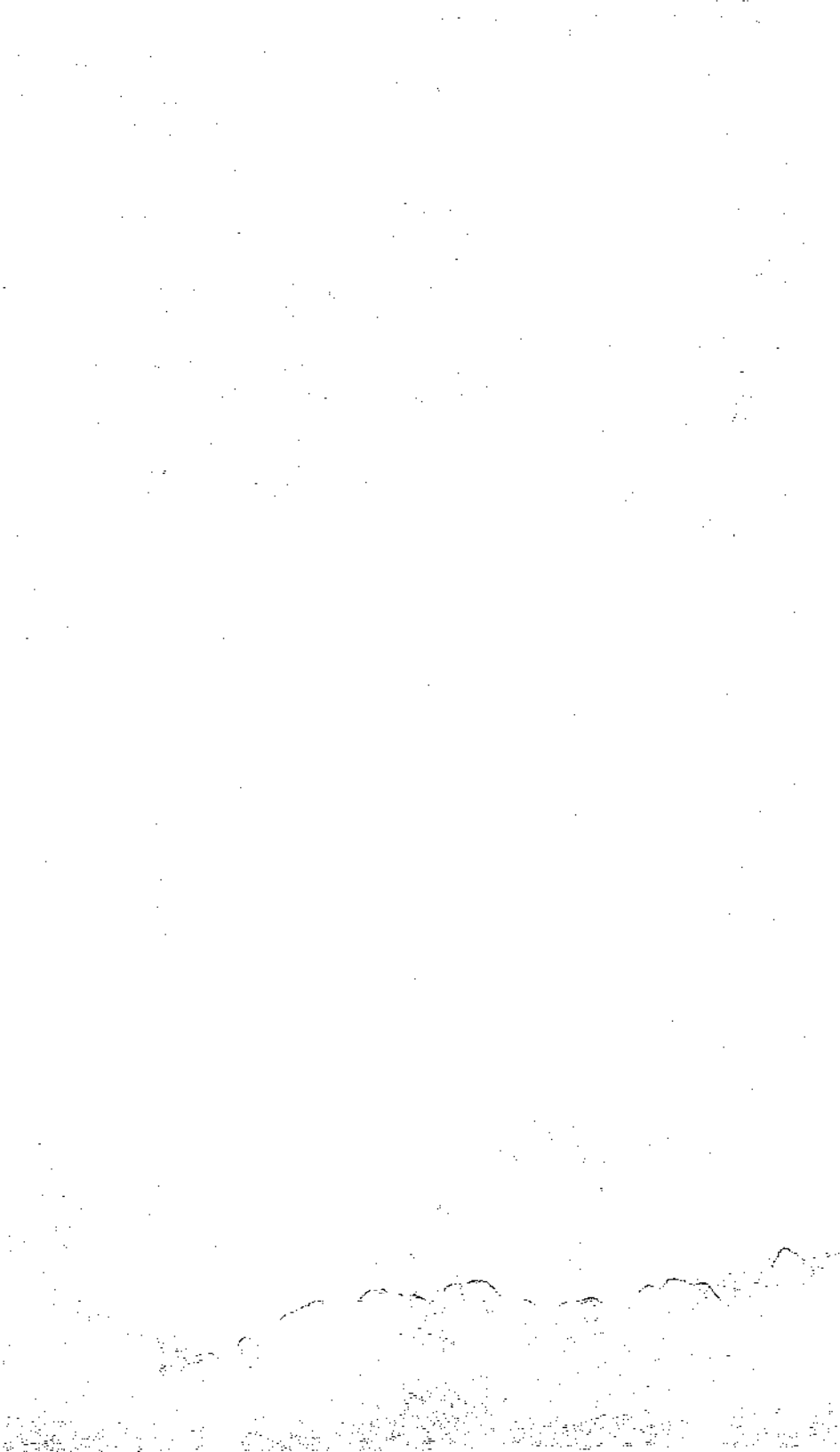
it means that the system loaded a segment register with an illegal value. This means that it is either a value not defined by the current descriptor table or the protection is such that the current user is not allowed to use that segment value or the code is trying to access out of the limits acceptable for that segment value as defined in the descriptor table.

If your system shuts down because of a general protection trap, reboot and note the time and circumstances, such as which users were logged on and what software they were using at the time of the error. If the error occurs repeatedly, it should be possible to locate the source of the problem and rectify the situation.

19.17 Mapping a Bad Track

Bad tracks on the hard disk are mapped during the XENIX installation procedure. This allows XENIX to avoid those areas of the disk that cannot be read or written. However, hard disks can develop bad tracks after XENIX is installed and running. If this occurs, the **badtrk**(ADM) utility should be run by the super user (root) enabling XENIX to avoid the new bad track(s). Be sure to run **badtrk** in non-destructive mode to save the data on your hard disk. **badtrk** must be run in single-user mode. Use **/etc/shutdown su** to enter single-user mode from multi-user mode (see **shutdown**(ADM)).

badtrk is a menu-driven utility for viewing, adding, or deleting entries to the bad track table. See **badtrk**(ADM) in this guide for more on its options and their use.



This page intentionally left blank.



Index

A

accept command 15-11
Accounting, record of processes 5-14
acctcom command, display 5-14
accton command
 start process accounting 5-14
adb
 kernel patches 19-4
 loading 19-4
Administration, process accounting 5-14
AUTO, gettydefs file 14-11

B

BACKSPACE key 1-4
backup account 6-2
Backup system. *See* Filesystem
Bidirectional port. *See* UUCP, dial-in/dial-out
/bin directory 13-1
Block
 arrangement on disk 13-8
 defined 4-16
 number 13-8
 ownership 4-17
 size 13-7
Bootstrap program 3-1
BREAK key 1-4
Building a Remote Network System. *See* UUCP
Bus cards 17-1

C

C program
 compilation header files 13-5
 library files 13-4
Cartridge tape
 configuration 16-1
 drive 16-1
 /etc/default/files 16-3
 formatting 16-5
 maintenance 16-4
chmod command
 permissions change 5-6

chmod command (*continued*)
 special file protection 5-4
Command
 /etc/rc file, appending to 10-14
 location
 /bin directory 13-1
 /usr/bin directory 13-5
 .login file, appending to 10-15
 .profile file, appending to 10-15
Computer
 See also Micnet network
 Micnet network, connection test 12-17
Console
 keyboard lock-up problem 19-6
 serial console 14-4
Context Indicator, sysadmsh screen area 2-3
Copy floppy disks 16-7
Core file, described 4-19
Ctrl-\ key 1-4
Ctrl-H key 1-4
Ctrl-Q key 1-4
Ctrl-S key 1-4
Ctrl-U key 1-4

D

dd command, copying a floppy 16-9
Default 16-3
Description Line, sysadmsh screen area 2-3
/dev directory
 contents 13-2
 special file 5-3
Device special filenames 13-7
df command, display free space 4-16
dialer programs
 compiling 14-19
 using 11-41, 14-19
Dial-in, special password protection 5-12
Dial-in/Dial-out. *See* UUCP, dial-in/dial-out
dial_login, gettydefs file entry 14-11
Directory
 access permissions 5-4
 block usage 4-17
 location 4-18
 permissions 5-4
 removing 10-11
 search, size effect 4-21
 size consideration 4-21
Disable
 command 14-17

Index

Disable (*continued*)

lineprinters 14-17, 15-12

terminal 14-17

disable command 14-17

Disk

block number 13-8

block size 13-7

damage. *See* Filesystem, damage

free space. *See* Filesystem, space

gap number 13-8

security 5-2

usage 4-17

diskcp command 16-8

formatting a floppy 16-8

Display Area, sysadmsh screen area 2-4

DOS

compile XENIX programs for DOS 9-8

files

access to 9-7

use of 9-1

install XENIX on DOS system 9-4

removing partition 9-7

utilities

access to 9-7

use of 9-1

du command 4-17

E

Enable

command 15-12

lineprinters 15-12

enable command 15-12

Error messages 13-5

Error Messages, sysadmsh screen area 2-4

ESCAPE key 1-4

/etc directory contents 13-3

/etc/default directory, contents 13-4

/etc/default/archive file, contents 13-4

/etc/default/backup file, contents 13-4

/etc/default/boot file, contents 13-4

/etc/default/cron file, contents 13-4

/etc/default/dumpdir file, contents 13-4

/etc/default/filesys file, contents 13-4

/etc/default/format file, contents 13-4

/etc/default/login file, contents 13-4

/etc/default/lpd file, contents 13-4

/etc/default/micnet file, contents 13-4

/etc/default/mkuser file, contents 13-4

/etc/default/msdos file, contents 9-8, 13-4

/etc/default/passwd file, contents 13-4

/etc/default/restor file, contents 13-4

/etc/default/su file, contents 13-4

/etc/default/tar file 16-3, contents 13-4

/etc/default/usemouse file, contents 13-4

/etc/gettydefs file 14-9, changing the file 14-11

/etc/group file, modifying 10-6

/etc/motd file

contents 10-15, 13-3

free space reminder 4-15

modifying 10-13, 10-15

/etc/passwd file

user entry 10-5

user ID change 10-9

/etc/passwd file, contents 13-3

/etc/rc file

contents 10-14, 13-3

Micnet network startup 12-16

modifying 10-13, 10-14

/etc/systemid file

machine name 12-2

Micnet network startup 12-16

/etc/termcap file

contents 13-3

terminal 14-15

Execute permission 5-5

F

fdisk command

partition

hard disk 9-1

table 9-2

use of XENIX and DOS 9-1

File

access. *See* Permission

backups. *See* Filesystem, backups

block size 4-17

core file 4-19

damage. *See* Filesystem

data loss 4-21

hidden file, removing 19-12

inaccessibility 4-21

initialization file. *See* Initialization file

location 4-18

log file, clearing 4-19

lost file, restoring. *See* Filesystem

permissions 5-4

recovery, using backup 19-13

removing

unused files 4-15

repair 4-25

restoring 6-17

system. *See* Filesystem

- File (*continued*)
 - temporary
 - removing 4-19
 - time of last use 4-18
 - unused, removing 4-15
 - Filename, device special files 13-7
 - Filesystem
 - automatic check 4-27
 - backups 6-1
 - backup account 6-2
 - floppy disk labeling 6-16
 - frequency 6-1
 - listing procedure 6-16
 - media storage 6-10
 - restoring 6-17
 - scheduled 6-9
 - unscheduled 6-13
 - changing sizes 4-21
 - cleaning 3-2
 - copies 6-1
 - creating, fileys 6-25
 - damage
 - causes 4-21
 - repairing 4-21, 4-22
 - data loss 4-21
 - defined 4-1
 - display free space 4-16
 - expanding 4-20
 - free space 4-14
 - full 4-15
 - maintenance 4-14
 - mounting
 - initialization files 10-13
 - security 4-12
 - repair 4-22
 - restoring 6-20
 - root filesystem
 - described 4-1
 - restoring 19-13
 - scheduled backup 6-9
 - space
 - displaying free space 4-16
 - lack of 4-15
 - maintaining 4-15
 - restoring 19-13
 - unmounting, umount command 4-12
 - unscheduled backup 6-13
 - user mount privileges 5-13
 - find command 4-18
 - Floppy disk
 - block size 13-7
 - bootable floppy disk 16-13
 - copying 16-7
 - damage. *See* Filesystem, damage
 - Floppy disk (*continued*)
 - Emergency Boot Floppy 16-13
 - Micnet file, saving 12-13
 - root filesystem disk 16-13
 - security 5-2
 - Free space. *See* Filesystem, space
 - fsck command 4-22
- ## G
- Gap number 13-8
 - getty
 - c option 14-13
 - program 14-9
 - gettydefs file 14-9
 - alternate login programs 14-11
 - autologin 14-11
 - changing the file 14-11
 - Group
 - access to 10-7
 - creating 10-6
 - defined 10-6
 - ID 10-3, 10-6
 - changing 10-8
 - name 10-3, 10-6
 - number 10-3
 - permissions 5-4
- ## H
- haltsys command 3-5
 - Hard disk
 - adding another 4-2
 - block size 13-7
 - damage. *See* Filesystem, damage
 - mounting 4-11
 - partitions
 - assigning 9-3
 - installing XENIX and DOS 9-5
 - tracks 9-1
 - two disks 4-2
 - installing user accounts 4-12
 - partitioning 9-6
 - using a mountable filesystem 4-12
 - Hayes modem with XENIX 14-29
 - Hidden file, removing 19-12
 - Highlighting a menu option 2-4
 - Home directory
 - removing 10-11

Index

Home directory (*continued*)
setting, initialization files 10-13
user account 10-1

I

Initialization file
contents 10-13
/etc/motd file. *See* /etc/motd file
/etc/rc file. *See* /etc/rc file
modifying 10-13
.profile. *See* .profile file
Internal memory, addition of 17-3
INTERRUPT key 1-4

J

Jumpers, bus card 17-1

K

Keyboard, described 1-4
kill command
stopping a runaway process 19-12
KILL key 1-4

L

l command, listing permissions 5-4
/lib directory contents 13-4
Lineprinter
accept command 15-11
adding, mkdev lp command 15-2
creating init device
mkdev lp command 15-9
disable command 15-12
dumb model interface 15-13
enable command 15-12
interface programs 15-13
lpmove command 15-10
lpsched program 15-8
lpshut command 15-8
lpstat command 15-8

Lineprinter (*continued*)
mkdev lp command 15-9
moving requests
lpmove command 15-10
parallel, mkdev lp command 15-9
polling operation 19-9
reject command 15-11
scheduler program, lpsched program 15-8
slow printing 19-9
solving problems 19-8
LOG file
contents 12-18
Micnet network, connection error 12-18
Log files 4-19
Login
group 10-7
name
Micnet network, entering 12-11
new user 10-2
sending mail 4-18
user account 10-1
programs
alternate, gettydefs file 14-11
dial_login 14-11
.login file
contents 10-15
modifying 10-15
lpmove command 15-10
lpsched program 15-8, 19-8
lpshut command 15-8
lpstat command 15-8
status of 15-8
stopping lpsched 15-8
lpshut command 15-8
lpstat command 15-8, 19-8

M

mail
command, message 4-18
Micnet network command 12-1
Mail
/usr/spool directory 13-5
Mailbox
removing 10-13
Maintenance
automated 11-52
manual 11-54
Memory
adding internal 17-3
parity errors 17-4
removing internal 17-4

- Menu option
 - highlighting 2-4
 - sysadmsh 2-4
- Menu Line, sysadmsh screen area 2-3
- Message, system-wide 4-16
- Message of the day file. *See* /etc/motd file
- Micnet network
 - alias
 - described 12-7
 - entry 12-12
 - forward 12-7
 - group, creating 12-7
 - machine 12-7
 - preparing 12-7
 - standard 12-7
 - composition 12-1
 - computer
 - connection test 12-17
 - machine name 12-2
 - connecting 12-17
 - /etc/systemid file
 - machine name 12-2
 - system startup 12-16
 - file
 - copying to computers 12-14
 - creating 12-8
 - modifying 12-20
 - restoring 12-14
 - saving 12-13
 - transferring 12-8
 - forward alias 12-7
 - group alias, creating 12-7
 - handshake message 12-18
 - install option 12-8
 - linear topology, described 12-2
- LOG file
 - connection error 12-18
 - contents 12-18
- machine
 - alias 12-7
 - name
 - choice 12-2
 - file entry 12-9
 - saving 12-2
- mail command 12-1
- modifying 12-20
- netutil program
 - information required 12-1
 - install option 12-8
 - network building 12-8
 - restore option 12-8, 12-14
 - save option 12-8, 12-13
 - start option 12-16
 - stop option 12-19
- Micnet network (*continued*)
 - planning 12-1
 - restore option 12-8, 12-14
 - save option 12-8, 12-13
 - serial line
 - assignment 12-3
 - name entry 12-10
 - transmission speed 12-6
 - standard alias 12-7
 - star topology, described 12-2
 - start option 12-16
 - startup, procedure 12-16
 - stop option 12-19
 - stopping 12-19
 - testing 12-17
 - topology
 - map 12-3
 - types 12-2
 - transmission speed
 - assignment 12-6
 - file entry 12-11
- mkdev lp command 15-2, 15-9
- mkuser program
 - creating a user account 10-1
 - passwd 10-3
 - shell type 10-4
 - stopping 10-2
- mnt command
 - mounting filesystems 4-12, 5-13
- /mnt directory, mounted filesystems 13-5
- Modem
 - Automatic Dial Modem 11-39
 - configuring the Hayes Smartmodem 2400
 - 11-14
 - control 11-21
 - Devices file 11-38
 - Dialers file 11-42
 - dial-in/dial-out site 11-2
 - files 11-8
 - installation 11-11
 - local network switch 11-43
 - login sequence 11-29
 - send and receive calls 11-13
 - serial lines 11-9, 11-12
 - telephone line 11-1
 - testing 11-16, 11-58
 - troubleshooting 11-58
 - usage
 - available serial lines 14-18
 - Devices file 14-19
 - dialer programs 14-19
 - dialing in 14-24
 - dialing out with cu 14-19
 - Hayes settings 14-29

Index

Modem (*continued*)

- UUCP, use of modem under 11-11
 - variable rate 11-16
- Modes of operation, described 3-2
- more command 10-14
- Motherboard cards 17-1

Mouse

- bus card 18-1
- installation 18-1
- parallel 18-1
- serial 18-1

Multiscreen 14-1

- access 14-1
- primary monitor 14-1

N

netutil program. *See* Micnet network

New user 10-1

newgrp command 10-7

Normal operation

- mode 3-2
- stopping 3-4

O

Operating system, loading 3-1

Out-of-space message 19-13

P

Parallel lineprinter, mkdev lp command 15-9

Parity errors, memory 17-4

Partition

- hard disk partitions
 - assigning 9-3
 - deleting 9-7
 - DOS 9-6
 - fdisk command 9-2
 - installing XENIX and DOS 9-5
 - two hard disks 9-6
- table 9-2

passwd command 5-9

Password

- changing 5-9
- complexity, system access security 5-2
- dial-in lines 5-12

Password (*continued*)

- forgotten 19-12
- new user 10-3
- super user 1-3
- user account 10-1

Permission

- changing 5-6
- description 5-4
- directories 5-4
- execute permission 5-5
- fields 5-4
- files 5-4
- group permissions 5-4
- initial assignment 5-6
- levels 5-4
- no permission 5-5
- other permissions 5-4
- read permission 5-5
- search permission 5-5
- special files 5-4
- user permissions 5-4
- write permission 5-5

PID, stopping a runaway process 19-12

Printer. *See* Lineprinter

Printing scheduler. *See* lpsched program

Process

- runaway 19-11
- stopping 19-11
- .profile file
 - contents 10-15
 - modifying 10-13, 10-15
 - removing 10-13
 - setting the terminal type 14-16

Program

- runaway process 19-11
- starting, initialization files 10-13
- terminating, core file placement 4-19

Q

quot command, block ownership display 4-17

R

rcp command 12-1

Read permission 5-5

reject command 15-11

remote command 12-1

Remote Network System. *See* UUCP

Restoring, root filesystem 19-13
 rm command 19-12
 rmuser command
 limitations 10-13
 removing a user account 10-11
 stopping 10-13
 root
 directory
 backup 6-1
 contents 13-1
 filesystem, restoring 19-13
 super user login name 3-3
 symbol (/) 4-1
 rs-232 11-9, 11-12, 11-14

S

Scan Windows, Scan keys 2-12
 Screens, multiple *See* Multiscreen 14-1
 Search permission 5-5
 Serial
 console 14-4
 line
 adding terminals 14-5
 device special filenames 13-8
 Micnet network
 assignment 12-3
 name entry 12-10
 transmission speed 12-6
 Serial ports
 adding 14-2
 configuring 14-2
 Setting
 console 14-4
 terminal
 lines 14-9
 type 14-15, 14-16
 Shutdown
 command 3-4
 improper shutdown, file check 4-27
 shutdown command 3-4
 Slash (/), root symbol 4-1
 Special file
 protection 5-4
 security 5-3
 Starting the system 3-1
 Status Line, sysadmsh screen area 2-3
 Stopping the system 3-4
 Super user
 account 1-3
 exiting 3-4
 log in 3-3
 Super user (*continued*)
 login name (root) 3-3
 password 1-3
 restricted use 1-3
 secrecy 5-3
 precautions 3-4
 prompt (#) 3-4
 special file access 5-4
 Switching operating systems 9-3
 sysadmin program
 creating backups 6-9
 described 6-2
 files, restoring 6-17, 19-13
 filesystem, restoring 6-20
 performing unscheduled backup 6-13
 sysadmsh program
 archive 6-25
 backups 6-16
 Context Indicator 2-3
 Display Area 2-4
 Error Messages 2-4
 filesys, creating 6-25
 menu, options 2-3
 Menu Line 2-3
 Status Line 2-3
 System
 access security 5-2
 accounts 10-13
 administration directory 13-5
 administrator. *See* System administrator
 cleaning the filesystem 3-2
 disk storage 4-1
 inoperable system, restoring 19-13
 maintenance 1-1
 account 1-3
 mode 3-2
 stopping 3-5
 physical security 5-2
 problems, fixing 19-1
 reinstalling 19-13
 security 5-2
 starting 3-1
 stopping 3-4
 system-wide message 4-16
 System administrator
 backups 6-1
 duties 1-1
 file access 4-1, 5-4
 filesystem maintenance 4-14
 free space, maintaining 4-15
 initialization files, modifying 10-13
 Micnet network maintenance 12-1
 super user account 1-3
 system maintenance mode 3-2

Index

System administrator (*continued*)
 user account creation, maintenance 10-1
System-wide message 4-16

T

Tape drive
 configuration 16-1
 /etc/default files 16-3
 formatting 16-5
 maintenance 16-4
 using 16-1
tar 16-3
Temporary file, removing 4-19
TERM variable 14-15
Terminal
 adding 14-5
 disabling 14-17
 lines, setting 14-9
 lockup, runaway process 19-11
 nonechoing terminal 19-1
 settings, checking 14-13
 type
 initialization files 10-13
 setting terminal type 14-15, 14-16
Time
 -atime parameter, date of last use 4-18
/tmp directory contents 13-5
tty line. *See* Serial line
ttys file 11-22
 and UUCP 11-22

U

umount command 4-12
User
 account
 adding 10-1
 comments 10-4
 directory, removing 10-11
 file, removing 10-11
 .login file 10-15
 login name 10-2
 .profile file 10-15
 removing 10-11
 block ownership display 4-17
 changing the ID 10-9
 group 10-7
 ID 10-9

User (*continued*)
 login group 10-7
 new user 10-1
 password. *See* Password
 permissions. *See* Permission
/usr directory contents 13-5
/usr/spool/lp/model file 15-13
uucico program
 connecting 11-6
 debugging with 11-58
 file transfer 11-7
 link to remote computer 11-6
 master and slave modes 11-8
 protocol negotiation 11-7
 READ and WRITE options 11-32
 Systems file 11-7
 uucico daemon 11-51
 work files 11-1
UUCP
 ACU (Automatic Calling Unit) 11-26, 11-47
 11-58
 administration 11-50
 cabling 11-3, 11-10, 11-14
 chat script 11-24, 11-28
 correcting 11-29
 defined 11-24, 11-28
 escape sequences in 11-29
 expect/send pairs 11-28
 subexpect/subsend pairs 11-28
 terminator on send string 11-28
 configuration files 11-5
 configuring 11-17
 connecting a serial wire 11-10
 control files 11-18
 creating passwords 11-23
 cron program 11-50
 polling passive systems 11-52
 cu program 11-14, 11-29
 daemons 11-6
 data files 11-55
 debugging 11-58
 described 11-1
 Devices file 11-5, 11-38
 speed field 11-16
 Dialcodes file 11-46
 Dialers file 11-8, 11-40
 dial-in/dial-out 11-18
 dialing prefixes 11-24
 direct wire connection 11-9
 directories 11-5
 /usr/spool/uucp 11-5
 /usr/spool/uucppublic 11-5
 /usr//uucp 11-5
 disable command 11-10, 11-21

UUCP (*continued*)

- enable command 11-21, 11-22
- error messages 11-61
- /etc/systemid file 11-20
- /etc/ttyS 11-22
- execute files 11-6, 11-51, 11-55
- filesystem, access to 11-5, 11-30, 11-31, 11-33, 11-35
- getty program 11-22, 11-44
- linking micnet sites 11-50
- links 11-24
 - ACU 11-39
 - direct 11-39
- Local Area Networks 11-38, 11-39
- lock files 11-8, 11-53, 11-55
- log files 11-5, 11-53, 11-54
- login
 - entries 11-23
 - IDs 11-5, 11-31
 - general access 11-35
 - restricted 11-30, 11-31, 11-36
 - prompt 11-29
 - script 11-24, 11-28
- LOGNAME entry
 - Permissions file 11-30, 11-32
- MACHINE entry, Permissions file 11-30
- mail, with UUCP 11-50
- maintenance
 - automated 11-52
 - manual 11-54
- micnet 11-1, 11-50
- modem 11-2, 11-28
 - configuring the Hayes Smartmodem 2400 11-14
 - connecting a modem 11-13
 - control 11-21
 - dialing configuration 11-13
 - installing 11-11
 - pin connections 11-3
 - serial lines 11-9, 11-12
 - testing 11-16
 - variable 11-16
- node name 11-24, 11-58
- ownership of files 11-19
- passwd file 11-23, 11-33
- passwords 11-5
 - creating 11-23
 - security 11-36
- Permissions file 11-5, 11-7, 11-19, 11-30
 - combining entries 11-37
 - list of options 11-31
 - rules for entries 11-30
- polling 11-52
 - with uudemmon.poll 11-52

UUCP (*continued*)

- protocols 11-39
- public directory 11-5, 11-31, 11-33
- remote commands 11-5, 11-55
 - allowing in Permissions 11-34
- retry period 11-26
- rmail program 11-31, 11-34, 11-36
- rs-232 11-2, 11-10
 - null modem cable 11-10
 - pin connectors 11-10, 11-14
- sample transaction 11-7
- schedule for calling other systems 11-50
- security 11-19, 11-34
 - access defined by login ID 11-31
 - filesystem, access to 11-35
 - login
 - IDs 11-5, 11-31
 - passwords 11-31
 - passive systems 11-32
- serial line
 - disabling 11-21
 - modem connection 11-9
- serial lines 11-12
- shared dial-in/dial-out 11-44
- sitename, choosing 11-20
- spool directory 11-55
- systemid file 11-20, 11-21
 - creating 11-20
- Systems file 11-24
 - device field 11-24, 11-26
 - format of entries 11-24
 - phone number field 11-24
 - schedule field 11-24, 11-25
 - speed field 11-26
- TM. (temporary data file) 11-55
- troubleshooting 11-2, 11-57
- uucico program
 - connecting 11-6
 - debugging with 11-58
 - file transfer 11-7
 - master and slave modes 11-8
 - protocol negotiation 11-7
- uucico program. *See* uucico program
- uucp program 11-1, 11-5
- uudemmon. admin 11-53
- uudemmon.clean 11-53, 11-55
- uudemmon.hour 11-7, 11-50, 11-51
- uudemmon.poll 11-50, 11-52
- uuninstall program 11-17
- uulog 11-54
- uulog command 11-59
- uuname command 11-59
- uusched program 11-6, 11-50
- uutry program 11-17, 11-58, 11-59

Index

UUCP (*continued*)

- uux command 11-20
- uux program 11-1, 11-5, 11-34, 11-54
- uuxqt program 11-6, 11-9, 11-35, 11-36, 11-51, 11-55
- work files 11-51, 11-55
- uucp program 11-1, 11-5
- uudemon.admin 11-53
- uudemon.clean 11-53, 11-55
- uudemon.hour 11-7, 11-50, 11-51
- uudemon.poll 11-50, 11-52
- uulog 11-54
- uulog command 11-59
- uuname command 11-59
- uusched program 11-6, 11-50
- uutry program 11-17, 11-58, 11-59
- uux command 11-20
- uux program 11-1, 11-5, 11-34, 11-54
- uuxqt program 11-6, 11-9, 11-35, 11-51, 11-55

W

- wall command 4-16
- Windows, Scan Window 2-12
- Write permission 5-5

X

XENIX

- keyboard 1-4
- removing partition 9-7
- xenix file 13-1

This page intentionally left blank.



Contents

System Administration (ADM)

intro	Introduction to system administration.
acctcom	Searches for and prints process accounting files.
accton	Turns on accounting.
adfnt	Formats SCSI hard disks.
alishash	Micnet alias hash table generator.
asktime	Prompts for the correct time of day.
autoboot	Automatically boot system.
badtrk	Disk flaws, scans for flaws and creates bad track table.
chroot	Changes root directory for command.
chsh	Changes a user's login shell entry in the password file.
cli	Clears inode.
config	Configures a XENIX system.
configure	XENIX configuration program.
consoleprint	Print <code>/usr/adm/messages</code> or any file to a serial printer attached to the printer port of a serial console.
custom	Installs specific portions of the XENIX System.
dial	Establish an outgoing terminal line connection.
divvy	Divides disk partitions.
dmesg	Displays the system messages on the console.
dparam	Displays/changes hard disk characteristics.
fdisk	Maintain disk partitions.
fdswap	Swaps default boot floppy drives.
fixperm	Correct or initialize file permissions and ownership.
fsave	Interactive, error-checking file system backup.
fsck	Checks and repairs file systems.
fsdb	File system debugger.
fssize	Prints or changes the name of a file system.
fsphoto	Performs periodic semi-automated system backups.
haltsys, reboot	Closes out the file systems and shuts down the system
hinstall	Places newly-created kernel in default location.
idleout	Logs out idle users.
install	Installation shell script.
ipcrm	Removes a message queue, semaphore set or shared memory ID.

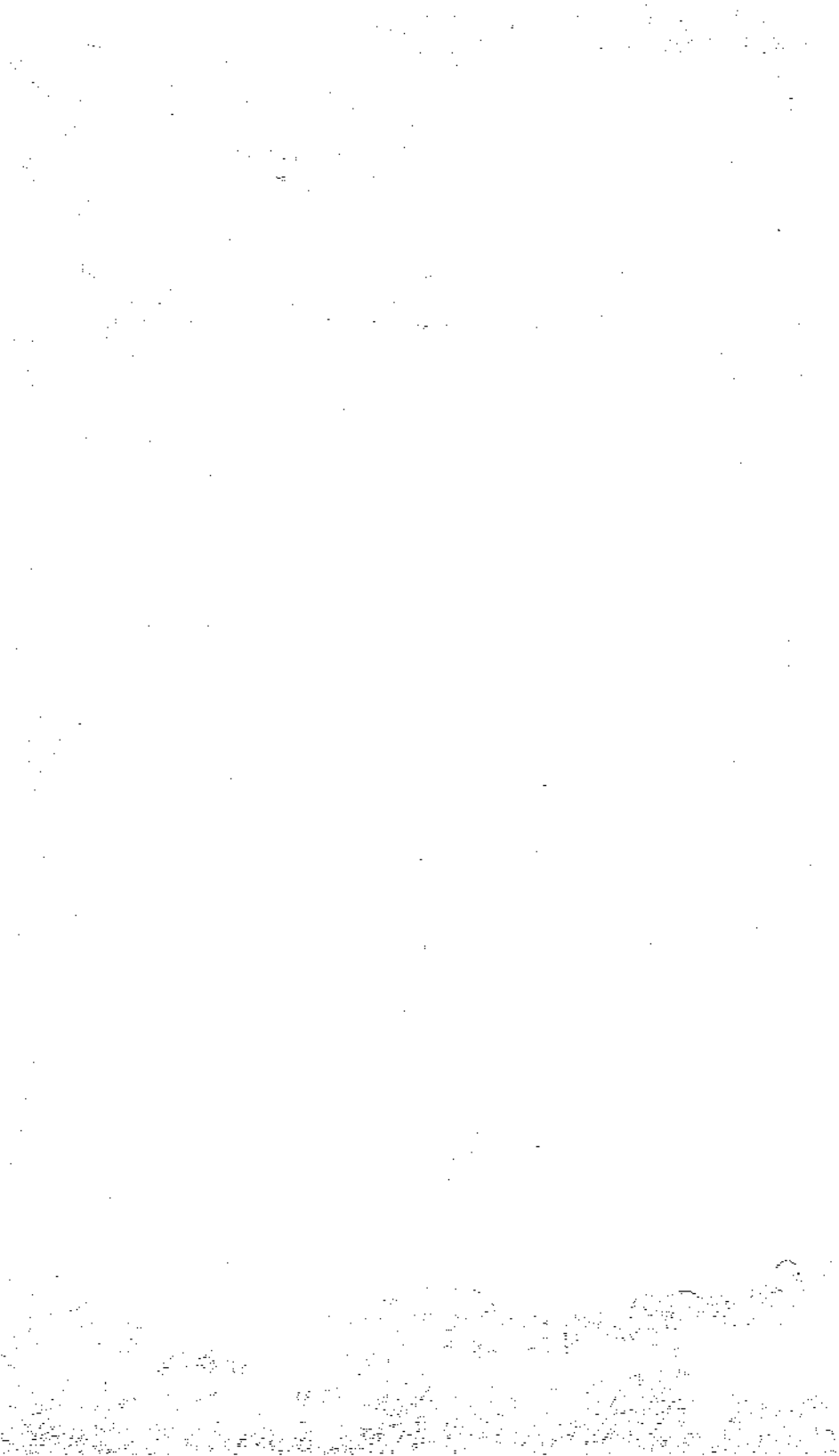
ipcs	Reports the status of inter-process communication.
ips, isbs, ipbs	IMAGEN protocol handlers.
kbmode	Tests or configures keyboard support.
lpadmin	Configures the lineprinter spooling system.
lpinit	Adds, reconfigures and maintains lineprinters.
lpsched, lpshut,	
lpmove	Starts/stops the lineprinter.
makekey	Generates an encryption key.
mkdev	Calls scripts to add peripheral devices.
mkfs	Constructs a file system.
mkuser	Adds a login ID to the system.
mount	Mounts a file structure.
mvdir	Moves a directory.
ncheck	Generates names from inode numbers.
netutil	Administers the XENIX network.
pwdadmin	Performs password aging administration.
rmuser	Removes a user account from the system.
runbig	Runs a command that may require more memory than normal.
schedule	Database for automated system backups.
setclock	Sets system real time clock.
setmnt	Establishes /etc/mnttab table.
settime	Changes the access and modification dates of files.
shutdown	Terminates all processing.
sync	Updates the super-block.
sysadmin	Performs file system backups and restores files.
sysadmsh	Menu driven system administration utility.
telinit, mkinittab	Alternative method of turning terminals on and off.
umount	Dismounts a file structure.
uuccheck	Checks the uucp directories and permissions file.
uucico	File transport program for the uucp system.
uuclean	UUCP spool directory clean-up.
uucinstall	Administers UUCP control files.
uusched	The scheduler for the uucp file transport program.
uutry	Tries to contact remote system with debugging on.
uuxqt	Executes remote command requests.
wall	Writes to all users.

Name

intro - Introduction to system administration commands.

Description

This section contains the commands that are used to administrate and maintain the XENIX operating system. These commands are largely root-only, meaning that they can only be executed by the super-user (root).



Name

acctcom - Searches for and prints process accounting files.

Syntax

acctcom [[options][file]] . . .

Description

acctcom reads *file*, the standard input, or **/usr/adm/pacct**, in the form described by *acct*(F) and writes selected records to the standard output. Each record represents the execution of one process. The output shows the COMMAND NAME, USER, TTYNAME, START TIME, END TIME, REAL (SEC), CPU (SEC), MEAN SIZE(K), and optionally, F (the *fork/exec* flag: **1** for *fork* without *exec*) and STAT (the system exit status).

The command name is prepended with a # if it was executed with super-user privileges. If a process is not associated with a known terminal, a ? is printed in the TTYNAME field.

If no *files* are specified, and if the standard input is associated with a terminal or **/dev/null** (as is the case when using **&** in the shell), **/usr/adm/pacct** is read, otherwise the standard input is read.

If any *file* arguments are given, they are read in their respective order. Each file is normally read forward, i.e., in chronological order by process completion time. The file **/usr/adm/pacct** is usually the current file to be examined; a busy system may need several files, in which case all but the current file will be found in **/usr/adm/pacct?**. The *options* are:

- b** Reads backwards, showing latest commands first.
- f** Prints the *fork/exec* flag and system exit status columns in the output.
- h** Instead of showing mean memory size, it shows the fraction of total available CPU time consumed by the process during its execution. This "hog factor" is computed as:
$$\text{(total CPU time)} / \text{(elapsed time)}$$
- i** Prints columns containing the I/O counts in the output.
- k** Instead of memory size, shows total kcore-minutes.

- m** Shows mean core size (the default).
- r** Shows CPU factor (user time/(system-time + user-time).)
- t** Shows separate system and user CPU times.
- v** Excludes column headings from the output.
- l *line*** Shows only processes belonging to terminal */dev/line*.
- u *user*** Shows only processes belonging to *user* that may be specified by a user ID, a login name that is then converted to a user ID, a # which designates only those processes executed with super-user privileges, or ? which designates only those processes associated with unknown user IDs.
- g *group*** Shows only processes belonging to *group*. The *group* may be designated by either the group ID or group name.
- d *mm/dd*** Any *time* arguments following this flag are assumed to occur on the given month and day, rather than during the last 24 hours. This is needed for looking at old files.
- s *time*** Shows only those processes that existed on or after *time*, given in the form *hr:min:sec*. The *:sec* or *:min:sec* may be omitted.
- e *time*** Shows only those processes that existed on or before *time*. Using the same *time* for both **-s** and **-e** shows the processes that existed at *time*.
- n *pattern*** Shows only commands matching *pattern* that may be a regular expression as in *ed* (C) except that + means one or more occurrences.
- H *factor*** Shows only processes that exceed *factor*, where factor is the "hog factor" as explained in option **-h** above.
- I *number*** Shows driver processes transferring more characters than the cutoff *number*.
- O *time*** Shows only those processes with operating system CPU time that exceeds *time*.
- C *time*** Shows only those processes that exceed *time* (the total CPU time).

Multiple options have the effect of a logical AND.

Files

/etc/passwd

/usr/adm/pacct

/etc/group

See Also

accton(ADM), ps(C), su(C), acct(S), acct(F), utmp(F)

Notes

acctcom only reports on processes that have terminated; use *ps*(C) for active processes.



Name

accton - Turns on accounting.

Syntax

accton [file]

Description

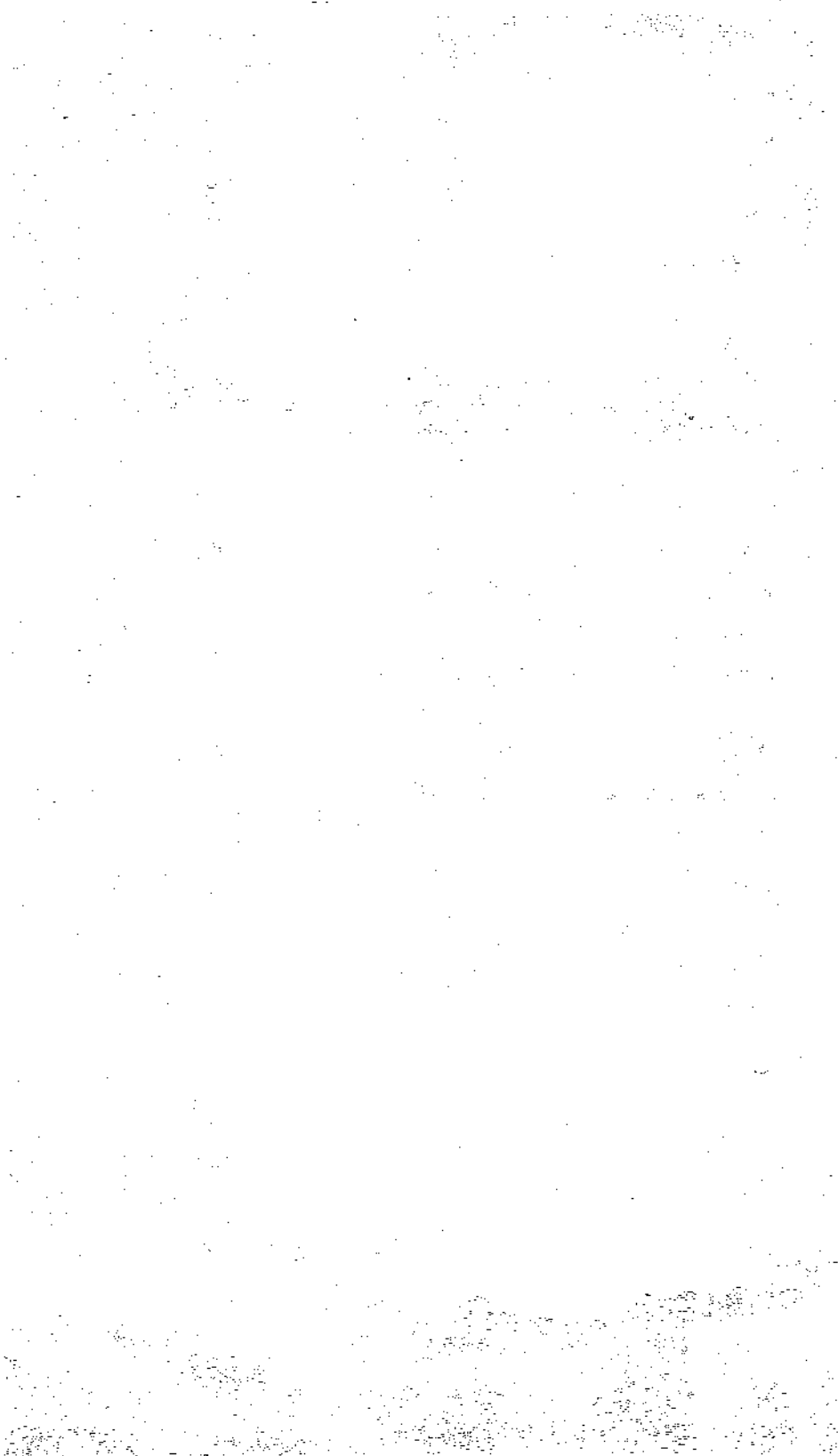
accton turns on and off process accounting. If no *file* is given then accounting is turned off. If *file* is given, the kernel appends process accounting records. (See *acct* (S) and *acct* (F)).

Files

/etc/passwd	Used for login name to user ID conversions
/usr/adm/pacct	Current process accounting file
/usr/adm/sulogin	Super-user login history file
/etc/wtmp	Login/logout history file

See Also

acctcom(ADM), acct(S), acct(F), su(C), utmp(F)



Name

adfmt - Formats SCSI hard disks.

Syntax

/etc/adfmt device_name

Description

The **adfmt** command issues a **format** command to the SCSI disk *device_name*. *device_name* should be the character-special device representing the whole SCSI disk, for example, */dev/rhd00*.

Notes

<p>This utility is not applicable to all hardware/software configurations and may not be included in your distribution.</p>

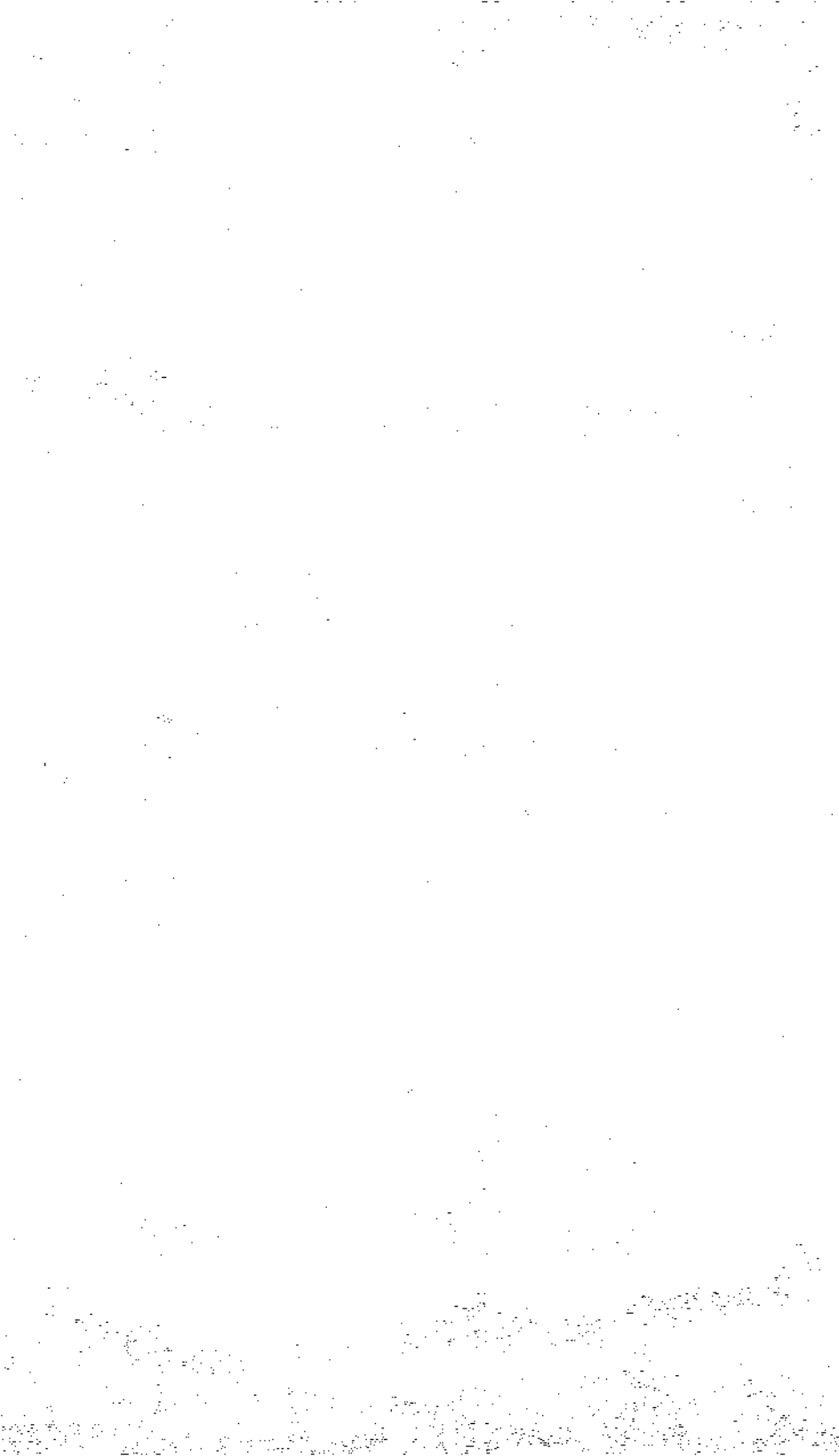
SCSI disks with embedded controllers are formatted as part of the manufacturing test procedure. Using **adfmt** on these disks is unnecessary.

Files

/dev/rhd?0

See Also

scsi(HW)
hd(HW)



Name

aliashash - Micnet alias hash table generator.

Syntax

```
aliashash [ -v ] [ -o output-file ] [ input-file ]
```

Description

The *aliashash* command reads the *input-file* and generates an *output-file* containing a hash table of alias definitions for a Micnet network. The *input-file* must name a file containing alias definitions in the form described for the **aliases** file (see *aliases*(M)). If the **-o** option is not used to specify an *output-file*, the command creates a file with the same name as the *input-file* but with **.hash** appended to it. If no *input-file* is given, the command reads the file named **/usr/lib/mail/aliases** and creates the file named **/usr/lib/mail/aliases.hash**.

If invoked with the **-v** option, the command lists information about the hash table.

The *output-file* will contain both the alias definitions given in the *input-file* and the new hash table. The hash table appears at the beginning of the file and is separated from the alias definitions by a blank line. The hash table has three or more lines. The first line is:

```
#<hash>
```

The second line has 4 entries: the bytes per table entry, the maximum number of items per hash value, the number of entries in the table, and the offset (in bytes) from the beginning of the file to the beginning of the alias definitions.

The next lines (up to the end of the hash table) contain the hash table entries. Each line has 8 entries (separated by spaces) and each entry has 2 fields. The first field (1 byte) is a checksum (represented as a printable character); the second field is a pointer (in bytes) to the alias definition. The pointer is represented as a hexadecimal number with leading blanks if necessary and is always relative to the start of the definitions.

The *aliashash* command is normally invoked by the **install** option of the *netutil* command. If the alias definitions of a network must be changed, the definitions in the **aliases** file should be changed and a new **aliases.hash** file created using the *aliashash* command. The new **aliases.hash** file must then be copied to all other computers in the network.

Files

```
/usr/lib/mail/aliashash  
/usr/lib/mail/aliases  
/usr/lib/mail/aliases.hash  
/usr/lib/mail/maliases.hash
```

See Also

aliases(M), netutil(ADM)

Warning

Do not use the *aliashash* command to create the **aliases.hash** file while the network is running. If necessary, create a temporary output file, **aliases.hash-**, using the **-o** option, then enter:

```
mv aliases.hash- aliases.hash
```

This will prevent disruption of the network.

Name

asktime - Prompts for the correct time of day.

Syntax

`/etc/asktime`

Description

This command prompts for the time of day. You must enter a legal time according to the proper format as defined below:

`[[yy]mmdd]hhmm`

Here the first *mm* is the month number; *dd* is the day number in the month; *hh* is the hour number (24-hour system); the second *mm* is the minute number; *yy* is the last 2 digits of the year number and is optional. The current year is the default if no year is mentioned.

Examples

This example sets the new time, date, and year to “11:29 April 20, 1985”.

```
Current system time is Wed Nov 3 14:36:23 PST 1985
Enter time ([yymmdd]hhmm): 8504201129
```

Diagnostics

If you enter an illegal time, *asktime* prompts with:

Try again:

Notes

asktime is normally performed automatically by the system startup file `/etc/rc` immediately after the system is booted; however, it may be executed at any time. The command is privileged, and can only be executed by the super-user.

Systems which autoboot will invoke *asktime* automatically on reboot. On these systems, if you don't enter a new time or press return within 1 minute of invoking *asktime* , the system will use the time value it has. If RETURN alone is entered, the time is unchanged.

Name

autoboot - Automatically boots the system.

Description

The system can be set up to go through the *boot* stages automatically (as defined in **/etc/default/boot** when the computer is turned on (booted), provided no key is pressed at the *boot(HW)* prompt.

If boot times out and LOADXENIX=YES, then XENIX is passed the word "auto" in its boot string and *init(M)*, *fsck(ADM)*, and *asktime(ADM)* are passed an **-a** flag.

In addition, the TIMEOUT entry can be set to specify the number of seconds to wait before timing out.

The *autoboot* procedure checks the file **/etc/default/boot** for the following instructions on autobooting:

LOADXENIX=YES or NO	Whether or not <i>boot(HW)</i> times out and loads XENIX. <i>boot</i> looks for this variable in the /etc/default/boot file on its default device.
FSCKFIX=YES or NO	Whether or not <i>fsck(ADM)</i> fixes any root system problems by itself. If the variable is set at YES, then <i>fsck(ADM)</i> is run on the root filesystem with the -rr flag.
MULTIUSER=YES or NO	Whether or not <i>init(M)</i> invokes <i>sulogin</i> or proceeds to multiuser mode.
PANICBOOT=YES or NO	Whether or not the system reboots after a panic(). This variable is read from /etc/default/boot by <i>init</i> .
ROONLYROOT=YES or NO	Whether or not the root filesystem is mounted <i>readonly</i> . This must be used only during installation, and not for a normal boot. It will effectively prevent writing to the filesystem.
DEFBOOTSTR= <i>bootstring</i>	Set default bootstring to <i>bootstring</i> . This is the string used by boot when the user presses <RETURN> only to the "Boot:" prompt, or when boot times out.

SYSTTY=*x*

If *x* is one (1), the system console device is set to the serial adapter at COM. If *x* is zero (0), the system console is set to the main display adapter.

TIMEOUT=*n*

where *n* is the number of seconds to timeout at the "Boot:" prompt before booting the kernel (if **LOADXENIX=YES**). If **TIMEOUT** is unspecified, defaults to one minute.

If either the **/etc/default/boot** file or the variable needed cannot be found, the variable is assumed to be NO. However, if the filesystem cannot be found, **PANICBOOT** is YES.

The **/etc/default/boot** file is shipped with the following default configuration:

```
LOADXENIX=YES
FSCKFIX=YES
MULTIUSER=YES
PANICBOOT=NO
```

A scratch file is needed by *fsck* to check large filesystems. The user is informed during the installation of XENIX if the system needs a scratch file to *fsck* the root filesystem. If necessary, the installation procedure creates the filesystem **/dev/scratch** to write the *fsck* temporary file. *fsck* uses the file named on the **/etc/default/boot** line:

SCRATCH=

as a scratch file. If the installation procedure creates the scratch filesystem, the entry in the **/etc/default/boot** is automatically made.

SCRATCH need only be specified if the root filesystem is large enough to need a temporary file. If a file is specified, it is always passed to *fsck* when checking the root filesystem, even if the system is *booted* manually. The only exception is the first time XENIX is booted from the hard disk, when the user must specify the scratch file. The file specified as **SCRATCH** must not be on the filesystem being checked by *fsck*. **SCRATCH** also can not be on an unmounted filesystem.

If the XENIX mail system, *mail(C)*, is installed on the system, the output of the boot sequence is mailed to *root*. Otherwise, the system administrator should check the file **/etc/bootlog** for the boot sequence output. The output of *fsck(ADM)* is temporarily saved in the file **/dev/recover** before it is moved to **/etc/bootlog** and finally may be sent to the system administrator via *mail*.

Other *boot* options which take affect during *autoboot* are documented on the *boot*(HW) manual page.

Files

<i>/etc/bootlog</i>	<i>boot</i> output log for autobooting systems
<i>/etc/default/boot</i>	boot information file
<i>/etc/rc</i>	instructions for entering multi-user mode, includes mounting and checking additional file systems
<i>/etc/sulogin</i>	executed at start-up, prompts the user to press Ctrl-d for multiuser mode or to enter the root password for maintenance mode
<i>/dev/recover</i>	allows saving of <i>fsck</i> output
<i>/dev/scratch</i>	temporary <i>fsck</i> file for large filesystems

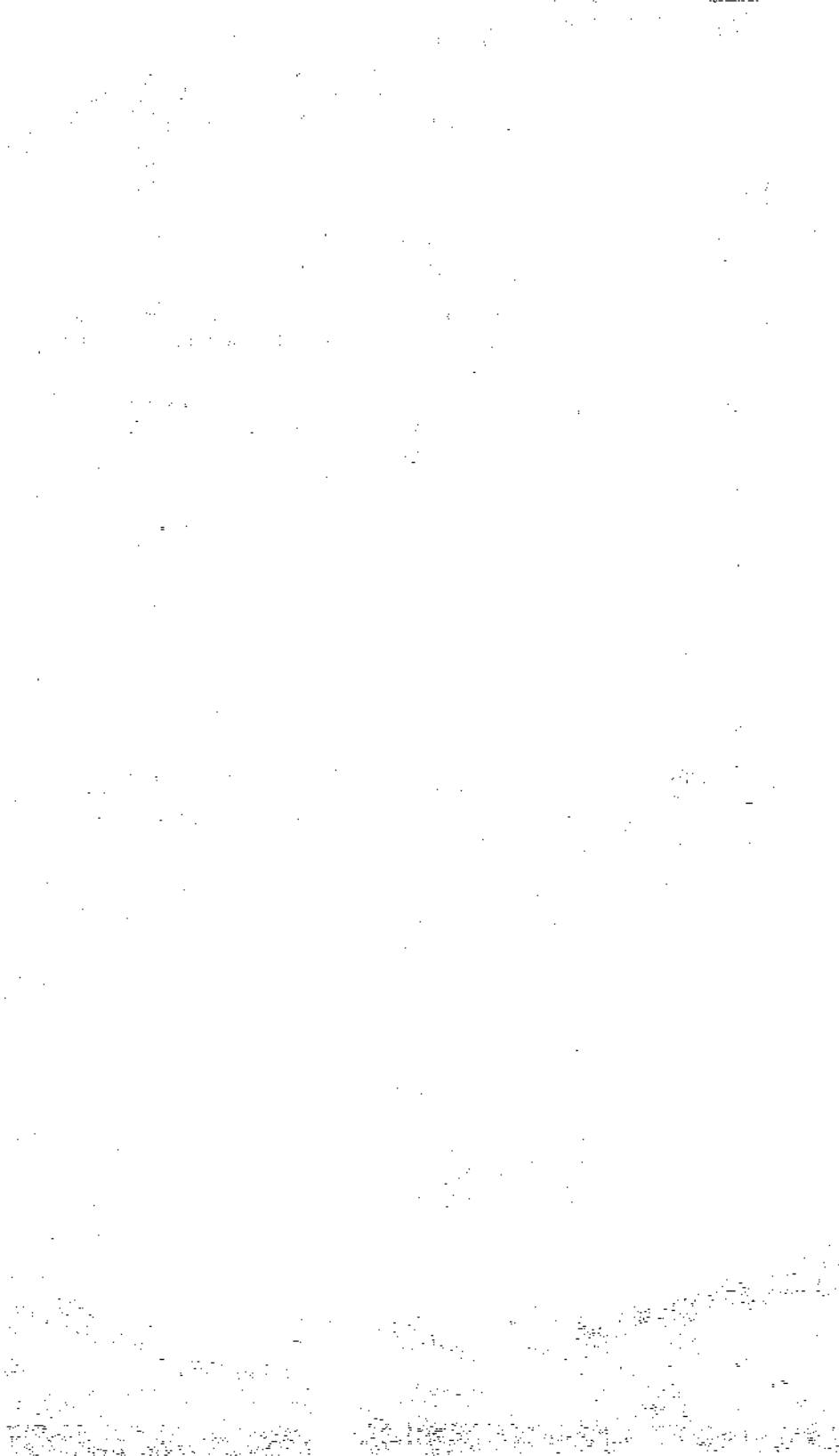
See Also

boot(HW), *fsck*(ADM), *init*(M)

Notes

The utilities invoked during the *boot* procedure are passed the *-a* flag and time out only when the system *autoboots*. For example, *asktime*(ADM) times out after 30 seconds when the system *autoboots*, but waits for a response from the user any other time it is invoked.

The previous *boot* modes of AUTO=CLEAN, DIRTY, NEVER have been retained for backwards compatibility, but are ignored if any of the newer modes are present.



Name

badtrk - Scans fixed disk for flaws and creates bad track table.

Syntax

```
badtrk [-e] [-s qtdn] [-f /dev/rhd*]
```

Description

Used chiefly during system installation, *badtrk* scans the media surface for flaws, creates a new bad track table, prints the current table, and adds and deletes entries to the table.

WARNING: The **-e** flag should not be invoked by the user. It is called by *hdinit* during installation to change the space allocated for bad tracks. Use of the **-e** flag at any other time may restructure the hard disk, rendering much of the information stored on it unusable.

To use *badtrk*, you must be in single user mode. (See *shutdown*(ADM)). To address the active XENIX partition on your *primary* fixed disk, enter:

```
badtrk -f /dev/rhd0a
```

To address the active XENIX partition on your *secondary* fixed disk, enter:

```
badtrk -f /dev/rhd1a
```

Usage

When *badtrk* is executed, the program first displays the main menu:

1. Print Current Bad Track Table
2. Scan Disk (You may choose Read-Only or Destructive later)
3. Add Entries to Current Bad Track Table by Cylinder/Head Number
4. Add Entries to Current Bad Track Table by Sector Number
5. Delete Entries Individually From Current Bad Track Table
6. Delete All Entries From Bad Track Table

Enter your choice or 'q' to quit:

You are prompted for option numbers, and, depending upon the option, more information may be queried for later.

A bad track table (option '1') might look like this:

Defective Tracks

	Cylinder	Head	Sector Number(s)
1.	190	3	12971-12987

Press <RETURN> to continue.

Option "2" scans the disk for flaws. If *badtrk* thinks changes may have been made to your bad track table since entering *badtrk* or updating your table, you will be asked if you want to update the device with the new table before scanning. You should answer "y" to save your changes, 'n' if you don't want to save changes made up to this point. Next you are prompted for more information. After you respond to these prompts, *badtrk* begins its scan. You can interrupt a scan by typing "q" at any time. You are then prompted to continue the scan or return to the main menu.

As the program finds flawed tracks, it displays the location of each bad track. An example error message might be:

```
wd: ERROR : on fixed disk ctlr=0 dev=0/47 block=31434 cmd=00000020
status=00005180, sector = 62899, cylinder/head = 483/4
```

(You may see this kind of message if there is a read or write error during the scanning procedure.)

When the scan is complete, the main menu reappears. The program automatically enters any detected flaws in the bad track table.

If there are no entries in your bad track table and a scan does not reveal any flaws, but your disk is furnished with a flaw map, you should enter these flaws into the bad track table. Select either option "3" or "4" to add the entries. (See next paragraph.)

To add flaw locations to an existing bad track table, select either option "3" or option "4", depending upon the format of the flaw map furnished with your disk. Enter the defective tracks, one per line.

When you are satisfied that *badtrk* contains a table of the desired flaws, quit the *badtrk* program by entering "q" at the main menu.

If *badtrk* was invoked with the *-e* option (which should only occur when called by *hdinit*, during the XENIX installation procedure), if you are reinstalling and you have a valid disk division table, the following message is displayed prior to the *badtrk* menu:

This device contains a valid division table. Additional (non-root) filesystems can be preserved across this reinstallation. If you wish to be able to preserve these file systems later, you must not change the current limit of the bad track table, which is *n* bad tracks. Do you wish to leave it unchanged? <y/n>:/s+1

If you respond ‘y’, you will not be prompted later to enter a new limit for the size of your bad track table. You can add or delete entries, but you will not be allowed to increase the maximum number of bad tracks allocated. If you respond ‘n’ and the size of your bad track table is changed, your disk division table will be destroyed.

If you do not have a valid disk table or you selected ‘n’ when prompted, you are prompted for the number of bad tracks to allocate. There will be a recommended number of replacement tracks to allocate based on the number of known bad tracks plus an allowance for tracks that will go bad in the future. You should choose to allocate at least as many as the recommended number of replacement tracks. Make your choice carefully, because if you want to change this amount later, you will have to reinstall XENIX.

At this point, you are asked if you want to ‘update’. This is *badtrk*’s way of asking if any changes which were made should be saved. You should answer ‘y’ to save your changes, ‘n’ to leave the bad track table as it was when last updated.

Arguments

-f *name*

Opens the partition *name* and reads the bad track table associated with that partition. The default is */dev/rhd0a*.

-s *options*

Invokes *badtrk* non-interactively. Valid options for this flag are:

[q]uick
[t]horough
[d]estructive
[n]on-destructive

The -s flag takes two options at a time. Choose quick or thorough scan, and destructive or non-destructive scan.

Notes

This utility can only be used in single-user mode.

If a bad spot develops in the boot blocks or system tables at the very beginning of the fdisk partition, reinstallation is required.

Files

/etc/badtrk

Name

chroot - Changes root directory for command.

Syntax

chroot newroot command

Description

The given command is executed relative to the new root. The meaning of any initial slashes (/) in pathnames is changed for a command and any of its children to *newroot*. Furthermore, the initial working directory is *newroot*.

Notice that:

```
chroot newroot command >x
```

creates the file *x* relative to the original root, not the new one.

This command is restricted to the super-user.

The new root pathname is always relative to the current root even if a *chroot* is currently in effect. The *newroot* argument is relative to the current root of the running process. Note that it is not possible to change directories to what was formerly the parent of the new root directory; i.e., the *chroot* command supports the new root as an absolute root for the duration of the *command*. This means that “/..” is always equivalent to “/”.

See Also

chdir(S)

Notes

Exercise extreme caution when referencing special files in the new root file system.

command must be under *newroot* or *command* is reported:
command: not found



Name

chsh - Changes a user's login shell entry in the password file.

Syntax

chsh

Description

chsh is used to modify the login shell entry in **/etc/passwd**. *chsh* prompts for a login name, confirms the entry, and prompts before making the actual change to **/etc/passwd**.

Files

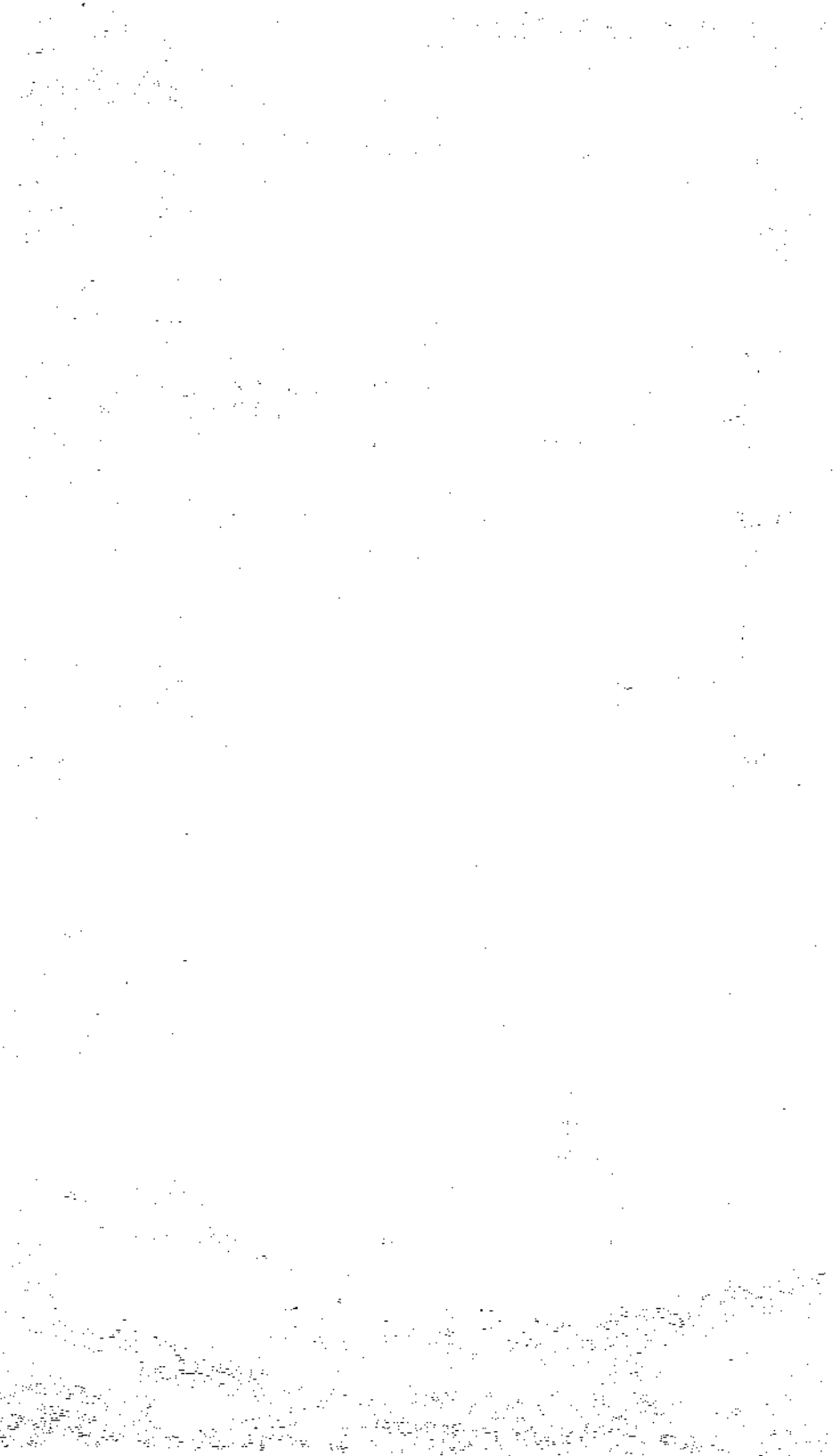
/etc/passwd

See Also

passwd(C), passwd(F)

Notes

Only the super user can invoke *chsh*.



Name

`clri` - Clears inode.

Syntax

`/etc/clri file-system i-number ...`

Description

`clri` writes zeros on the 64 bytes occupied by the inode numbered *i-number*. *File-system* must be a special filename referring to a device containing a file system. After `clri` is executed, any blocks in the affected file will show up as “missing” if the file system is checked with `fsck(ADM)`. Use `clri` only in emergencies and exercise extreme care.

Read and write permission is required on the specified *file-system* device. The inode becomes allocatable.

The primary purpose of this routine is to remove a file which, for some reason, does not appear in a directory. If you use `clri` to destroy an inode which does appear in a directory, track down the entry and remove it. Otherwise, when the inode is reallocated to some new file, the old entry will still point to this file. At that point removing the old entry will destroy the new file. The new entry will again point to an unallocated inode, so the whole cycle is likely to be repeated again and again.

See Also

`fsck(ADM)`, `ncheck(ADM)`

Notes

If the file is open, `clri` is likely to be ineffective.



Name

config - Configures a XENIX system.

Syntax

```
/usr/sys/conf/config [ -i ] [ -c file ] [ -s ] -m master dfile
```

Description

config takes a description of a XENIX system and generates compilable files that define the configuration tables for the various devices on the system.

Options include:

- m Specifies the name of the file that contains all the information regarding supported devices; */usr/sys/conf/master* is the standard name. This file is supplied with the XENIX system and should *not* be modified by the user. The *configure*(ADM) utility should be used to update */usr/sys/conf/master* and *dfile*.
- i Requests assembly-language output, instead of the default C language output.
- c Specifies the name of the configuration table file. *c.c* is the default names unless the *-i* option is given, in which case the default name is *c.asm*.
- s Specifies the name of the parameters file. *space.c* is the default name; if the *-i* option is used, the default name is *space.inc*.

dfile contains system device information and is divided into two parts. The first contains physical device specifications. The second contains system-dependent information. Any line with an asterisk (*) in column 1 is a comment. A standard *dfile* is provided as */usr/sys/conf/xenixconf*. The *configure*(ADM) utility should also be used to update */usr/sys/conf/xenixconf*.

All configurations are assumed to have a set of required devices, such as the system clock, which must be present to run XENIX. These devices *must not* be specified in *dfile*.

First Part of *dfile*

Each line contains two fields, delimited by spaces and/or tabs in the following format:

devname number

where *devname* is the name of the device, and *number* is the number (decimal) of devices associated with the corresponding controller. The device name can be any name given in part 1 of the **/usr/sys/conf/master** file, or any alias given in part 3 of the same file; *number* is optional, and if omitted, a default value which is the maximum value for that controller is used.

There are certain drivers that may be provided with the system that are actually *pseudo-device* drivers; that is, there is no real hardware associated with the driver. If the system has such drivers, they are described in section M of the *XENIX User's Reference*.

Second Part of *dfile*

The second part contains three different types of lines. Note that *all* specifications of this part *are required*, although their order is arbitrary.

1. **root/pipe** device specification

Two lines, each having three fields:

root	devname	minor
pipe	devname	minor

where *devname* is the name of the device, and *minor* is the minor device number (in octal). The device name can be any name given in part 1 of the **/usr/sys/conf/master** file, or any alias given in part 3 of the same file.

2. **swap** device specification

One line that contains five fields as follows:

swap	devname	minor	swplo	nswap
-------------	---------	-------	-------	-------

where *devname* is the name of the device, *minor* is the minor device number (in octal), *swplo* is the lowest disk block (decimal) in the swap area, and *nswap* is the number of disk blocks (decimal) in the swap area. The device name can be any name given in part 1 of the **/usr/sys/conf/master** file, or any alias given in part 3 of the same file.

3. *Parameter* specification

One or more lines, each having two fields as follows:

name	number
------	--------

where *name* is a tunable parameter name, and *number* is the desired value (in decimal) for the given parameter. Only names that have been defined in part 4 of the `/usr/sys/conf/master` file can be used; *number* overrides the default value for the given parameter. The following is a list of the available parameters:

buffers	Maximum number of external (mapped-out) buffers available to the kernel. If set to 0, <i>config</i> computes the optimum number for the system.
sabufs	Maximum number of internal (non-mapped) buffers available.
hashbuf	Maximum number of hash buffers.
inodes	Maximum number of inodes per file system.
files	Maximum number of open files per file system.
mounts	Maximum number of mounted file systems.
coremap	Maximum number of core map elements.
swapmap	Maximum number of swap map elements.
pages	Number of memory pages. On segmented systems such as the 286, this value should be 0.
calls	Maximum number of entries in the system timeout table.
procs	Maximum number of processes per system.
maxproc	Maximum number of processes per user.
texts	Maximum number of text segments per system.
clists	Maximum number of clists per system.
locks	Maximum number of file locks per system.
shdata	Maximum number of shared data segments per system.
timezone	Number of minutes difference between the local timezone and Greenwich Mean Time.
daylight	Daylight savings time in effect (1) or not in effect (0).
msgmap	Number of entries in message map.
msgmax	Maximum message size.
msgmnb	Maximum number of bytes in a message queue.
msgmni	Number of message queue identifiers.
msgtql	Number of message headers in the system.
msgssz	Number of bytes in message segments.
msgseg	Number of message segments.
semmap	Number of entries in semaphore map.
semmni	Number of semaphore identifiers.
semmnu	Number of undo structures in the system.
semmsl	Maximum number of semaphores per identifier.

semopm	Maximum number of operations per <i>semop</i> (S) call.
semume	Maximum number of undo entries per process.
semvmx	Maximum semaphore value.
semaem	Maximum value for "adjust on exit".
semmns	Number of semaphores in the system.
cmask	Default file creation mask for process 0.
maxprocmem	Maximum amount of memory available per process.
screens	Number of Multiscreens for the systems.
emaps	Maximum number of distinct eight-bit channel maps in the system.
nodename	The nodename of the system (as used by uucp (C) and other programs).
npbuf	The number of physical input/output buffers to allocate.
dmaexcl	This is set to 1 if only 1 DMA channel is usable at once, 0 otherwise.
sdslots	sdslots * shdata is the maximum number of simultaneous attaches to shared memory segments for the entire system.
memlim	A process may occupy up to this percent of user memory, plus the swap area it can occupy (which is constrained by swplim). This parameter is only valid on 80286-based machines.
swplim	A process may occupy up to this percent of swap area, plus the memory area it can occupy (which is constrained by memlim). This parameter is only valid on 80286-based machines.
maxbuf	Maximum possible number of buffer cache buffers.
shless	The number of shell-layer sessions.
shmmax	Specifies the maximum shared memory segment size. The default value is 131072.
shmmin	Specifies the minimum shared memory segment size. The default value is 1.
shmmni	Specifies the maximum number of shared memory identifiers system wide.
shmseg	Specifies the number of attached shared memory segments per process.
shmall	Specifies the maximum number of in-use shared memory text segments.
nqueue	The number of STREAMS queues to be configured.

nstream	The number of "Stream-head" (stdata) structures to be configured.
nstrpush	The maximum number of modules that may be pushed onto a Stream.
nstrevent	The initial number of Stream event cells to be configured.
maxsepgcnt	The number of additional pages of memory that can be dynamically allocated for event cells.
nmuxlink	The maximum number of multiplexer links to be configured.
strmsgsz	The maximum allowable size of the data portion of any STREAMS message.
strctlsz	The maximum allowable size of the control portion of any STREAMS message.
nblkn	The number of STREAMS data blocks and buffers to be allocated for each size class (n).
strlofrac	The percentage of data blocks of a given class at which low-priority block allocation requests are automatically failed.
strmedfrac	The percentage cutoff at which medium priority block allocations are failed (see strlofrac discussion above).
evqueues	Maximum number of open event queues system-wide.
evdevs	Maximum number of devices attached to event queues systemwide.
evdevsperq	Maximum devices per event queue.
katype	This is set to XT for XT-type keyboards and AT for AT-type keyboards.

Examples

Suppose you wish to configure a system with the following devices:

One HD disk drive controller with 1 drive
 One FD floppy disk drive controller with 1 drive

You must also specify the following parameter information:

root device is an HD (pseudo disk 3)
 pipe device is an HD (pseudo disk 3)
 swap device is an HD (pseudo disk 2)
 with a swplo of 0 and a nswap of 2300
 number of buffers is 50
 number of processes is 50
 maximum number of processes per user ID is 15
 number of mounts is 8
 number of inodes is 120

number of files is 120
 number of calls is 30
 number of texts is 35
 number of character buffers is 150
 number of swapmap entries is 50
 number of memory pages is 512
 number of file locks is 100
 timezone is pacific time
 daylight time is in effect
 number of entries in message map is 513
 maximum message size is 8192
 maximum number of bytes in a message queue is 16384
 number of message queue identifiers is 10
 number of message headers in the system is 40
 message segment size is 8
 number of message segments is 1024
 number of entries in semaphore map is 21
 number of semaphore identifiers is 10
 number of undo structures in the system is 60
 maximum number of semaphores per identifiers is 10
 maximum number of operations per *semop* call is 5
 maximum number of undo entries per process is 5
 maximum semaphore value is 32767
 maximum value for "adjust on exit" is 16384
 number of semaphores in the system is 40

The actual system configuration would be specified as follows:

```

hd      1
fd      1
root    hd      3
pipe    hd      3
swap    hd      2      0      2300
* Comments may be inserted in this manner
buffers 50
procs   150
maxproc 15
mounts  8
inodes  120
files   120
calls   30
texts   35
clists  150
swapmap          50
pages  (1024/2);
locks   100
timezone      (8*60)
daylight 1
msgmap (MSGSEG/2+1)
msgmax 8192
msgmnb 8192
msgmni 10
  
```



```

msgtql 40
msgssz 8
msgseg 1024
semmap (SEMMNS/2+1)
semmni 10
semmnu 20
semmsl 10
semopm 5
semume 5
semvmx 32767
semaem 16384
semmns 40

```

Files

/usr/sys/conf/master	default input master device table
c.c	default output driver configuration table file
space.c	default output resource configuration table file
c.asm	default driver configuration in assembly language
space.inc	default resource configuration in assembly language

See Also

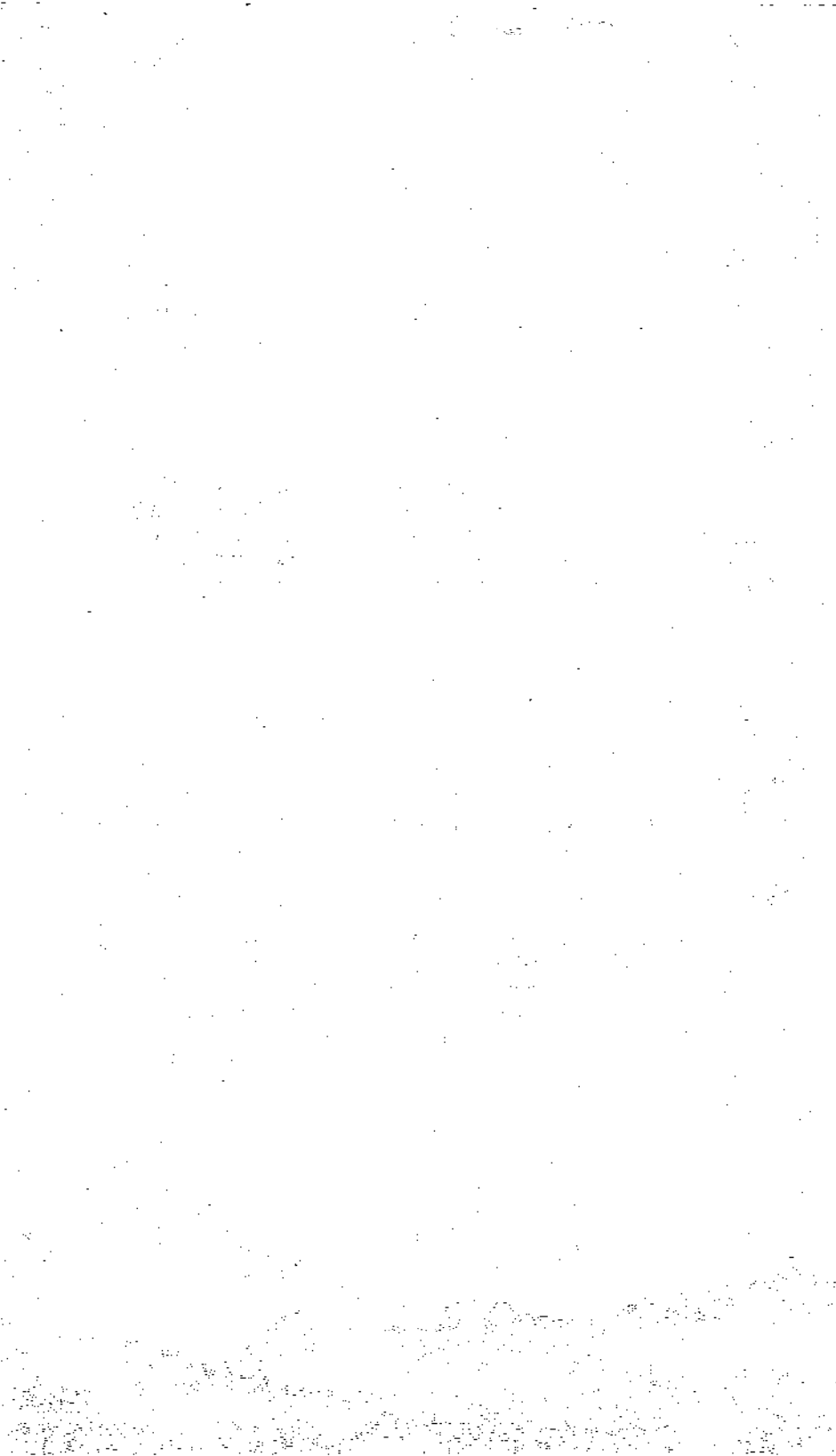
configure(ADM), master(F)

Diagnostics

Diagnostics are routed to the standard output and are self-explanatory.

Notes

The value on the right-hand side of a parameter specification must be a double-quoted character string, an integer, the name of another parameter defined within the **master(F)** file, or some arithmetical combination of integers and defined parameter names. Only the “+”, “-”, “*”, and “/” operators can be used in an arithmetical expression. Expressions are interpreted left-to-right: if operator precedence is in doubt, parenthesize.



Name

configure - **xenix** configuration program.

Syntax

configure [options] [*parm=val ...*]

Description

The *configure* program determines and alters different kernel resources. For end users, *configure* is easier than modifying the system configuration files directly. For device driver writers, *configure* avoids the difficulties of editing configuration files that have already been edited by an earlier driver configuration script.

Resources are modified interactively or with command-line arguments. Adding or deleting device driver components requires the command line options.

The next paragraphs discuss how to use *configure* interactively. Command line options are discussed in the "Options" section.

Interactive Usage

configure functions interactively when no options are given, or when **-f** is the only option specified on the command line.

When you invoke *configure* interactively, you first see a category menu that looks something like this:

1. Disk Buffers
2. Character Buffers
3. Files, Inodes, and Filesystems
4. Processes, Memory Management & Swapping
5. Clock
6. MultiScreens
7. Message Queues
8. Semaphores
9. Shared Data
10. System Name
11. Streams Data
12. Event Queues and Devices
13. Hardware Dependent Parameters

Select a parameter category to reconfigure by typing a number from 1 to 13, or type 'q' to quit:

To choose a category, enter its number, (e.g. "1" for "Disk Buffers") then press RETURN.

Each category contains a number of configurable resources. Each resource is presented by displaying its true name, a short description, and its current value. For example, for the "Disk Buffers" category you might see:

NBUF: total disk buffers. Currently determined at system start up:
NSABUF: system-addressable (near) disk buffers. Currently 10:
NHBUF: hash buffers (for disk block sorting). Currently 128:

To keep the current value, simply press RETURN. Otherwise, enter an appropriate value for the resource, then press RETURN. *configure* checks each value to make sure that it is within an appropriate range. If not, *configure* will warn you that the value is inappropriate and confirm that you wish to override the recommended value.

To exit from *configure* enter 'q' at the category menu prompt. If any changes are made, *configure* asks if it should update the configuration files with the changes. To keep the old configuration values, enter 'n' at this prompt, and no changes are made. Otherwise, enter 'y' and *configure* updates the required system configuration files. After *configure* has completed, the kernel is ready for linking.

To link the kernel, enter:

link_xenix

Linking may take a few minutes. After the kernel is linked, enter the following commands to place a copy of the new kernel (*xenix.new*) in the root directory and reboot the system:

```
cp /usr/sys/conf/xenix /xenix.new  
/etc/shutdown
```

Next, you see the boot prompt:

```
Boot  
:
```

To test the new kernel, enter the following at the boot prompt:

```
xenix.new
```

The system is now running the new kernel. When you are satisfied with the performance of the new kernel, enter the following command to install the new kernel on the hard disk:

/usr/sys/conf/hdinstall

The *hdinstall*(ADM) program backs up the old */xenix* and copies */usr/sys/conf/xenix* to */xenix*.

Remove *xenix.new* by entering the following command:

```
rm /xenix.new
```

Reboot the system to run the new kernel.

Options

The command line options are designed for writers of driver-installation shell scripts. You can configure drivers, remove driver definitions from the configuration files, and modify some driver attributes, all from the command line. There are also options for querying the current driver configuration.

configure uses the following options:

```
-a [func1 func2 ... ]
-d [func1 func2 ... ]
-b
-c
-d [func1 func2 ... ]
-f master_file [dfile ]
-g dev_name handler | dev_name
-j [prefix ] [NEXTMAJOR ]
-l priority_level
-m major
-n
-q
-r
-t
-v interrupt_vector [interrupt_vector2... ]
-w
-x
-y resource
```

-m, -b, and -c

These options are used to define which driver is being referenced. Following **-m** must be the major device number of the driver. If you are configuring a block driver, **-b** must appear; if you are configuring a character driver, **-c** must appear. Both are used when configuring a driver with both kinds of interfaces.

-a and -d

Each option is followed by a list of functions to add or delete, respectively. These are the names of the functions that appear within *bdevsw*[] or *cdevsw*[], as appropriate, plus the names of the

initialization, clock poll, halt and interrupt routines, if present, plus the names of the tty, stream, and tab structure pointers. *configure* enforces the rules that all of a driver's routines must have a common prefix, and that the prefix be 2-4 characters long.

- j When followed by a *prefix* used by a driver, the major device number is displayed. When followed by **NEXTMAJOR**, the smallest major device number is displayed.
- r This option forces a rewrite of the configuration files regardless of whether or not the command changed the configuration.
- v This option modifies the system notion of the vectors on which this device can interrupt. A device may interrupt on up to 4 vectors.
- l This sets the interrupt priority level of the device, which is almost always the same as the type of *spl()* call used: a driver that interlocks using *spl5()* almost always has an interrupt priority level of 5.
- q If the **-q** option is given, no *qswtch()* is possible after returning from the device interrupt. Use of this option in new drivers is not recommended.
- f The configuration is maintained in two data files, whose default names are *master* and *xenixconf*. The **-f** option can be used to specify alternate names. Note that if **-f** is the only option present, the program is still interactive.
- n If **-n** is present, the two configuration data files are modified, but no '.o' files are produced. This option is useful when configuring a driver package containing multiple drivers.
- w This option suppresses warning messages.
- x This dumps all the resource prompts known to *configure*. These reveal the name, description and current value of each parameter capable of being reconfigured. Category prompts are not dumped.
- y The **-y** option prints out the current value of the requested resource.
- t This option prints out nothing (except possibly error messages). However, it has a return value of 1 if a driver corresponding to the given combination of **-m**, **-b**, **-c** and options is already configured, and returns 0 if no such driver is present.
- g This option is used to add or remove graphics input (GIN) device handlers. Devices such as mice, bitpads, and keyboards may have handlers to turn their input data into "events." The **-g** flag may be given one argument that is interpreted as a device name. That GIN device is removed from the configuration files. If the **-g** flag has two arguments, the second is a handler for that device, and the

device is added to the files. If it was already present, its handler is updated and the user is informed. Multiple devices may be added or removed by specifying **-g** multiple times.

Setting Command-line Parameters

Any number of arguments can be given on the command line of the form *resource=value*. These arguments can be given at the same time as an add or delete driver request, but must follow all the driver-configuration arguments on the command line.

Some resources have values that are character strings. In this case their values must be enclosed within the characters `"`. The quotes are syntactically necessary for them to be used as C-language strings, and the backslashes protect the quotes from being removed by the shell.

Examples

Print out the current value of NCLIST:

```
configure -y NCLIST
```

Return 1 if character major device 7 and vector 3 are available:

```
configure -t -v 7 -m 3 -c
```

Add a clock-time polling and initialization routine to the already configured "foo" driver, a hypothetical character driver at major device #17:

```
configure -a foopoll fooinit -c -m 17
```

Delete the "foo" driver:

```
configure -m 17 -d -c
```

Add a new "hypo" driver, a block driver with a character interface. It absorbs 3 different interrupt vectors, at priority 6:

```
configure -a hypoopen hypoclose hyporead hypowrite hypoioctl\
hypostrategy hypotab hypointr -b -c -l 6 -v 17 42 49
```

Notes

Kernel Data Space Restrictions (286 only)

If the total size of all the allocated resources grows too large, the group will not fit within the kernel's 64k near data segment. You will not see messages about excessive size from *configure*, but you may see them from the linker when you attempt to link the kernel.

Files

/usr/sys/conf/master
/usr/sys/conf/xenixconf
/usr/sys/conf/config

See Also

master(F), config(ADM), event(M), hdinstall(ADM)

Name

consoleprint - print **/usr/adm/messages** or any file to a serial printer attached to the printer port of a serial console

Syntax

consoleprint [file]

Description

console(C) prints the file **/usr/adm/messages** to a printer attached to the printer port of a serial console. If a filename is specified, it is printed instead. *consoleprint* is normally run by a system administrator to get a hardcopy version of the system console messages.

This command uses the file **/etc/termcap**.

Files

/etc/termcap

See Also

lprint(C)

Notes

The only terminals currently supported with entries in **/etc/termcap** are the Tandy DT-100 and DT-1, and the Hewlett-Packard HP-92.

Terminal communications parameters (such as baud rate and parity) must be set up on the terminal by the user.



Name

custom - Installs specific portions of the XENIX System

Syntax

custom [-odt] [-irl [package]] [-m device] [-f [file]]

Description

With *custom* you can create a custom installation by selectively installing or deleting portions of the XENIX system. *custom* is executable only by the super-user and is either interactive or can be invoked from the command line with several options.

Files are extracted or deleted in *packages*. A package is a collection of individual files. Packages are grouped together in *sets*.

Three default *sets* are always available:

Operating System

Development System

Text Processing System

You can also install additional *sets*. You can list the available *packages* by using the *custom* command as described next.

Usage

To use *custom* interactively, enter:

```
custom
```

You see a list of sets. For example:

1. Operating System
2. Development System
3. Text Processing System
4. Add a Supported Product

The program prompts you to choose a set from which to work. If the data files for that set are not already installed on the hard disk, *custom* prompts you for the floppy which contains these data files and installs them. You may also see menu items for each product that has been previously added using the "Add a Supported Product" option. If you are adding a new product, you will be prompted for volume 1 of the new product distribution and *custom* will extract the product information necessary to support it.

When you select a valid set, you see a menu like this:

1. Install one or more packages
2. Remove one or more packages
3. List the files in a package
4. Install a single file
5. Select a new set to customize
6. Display current disk usage
7. Help

When you enter a menu option, you are prompted for further information. This is what the options prompt, and what action occurs:

1. Install

Prompts for one or more package names.

Calculates which installation volumes (distribution media) are needed, then prompts for the correct volume numbers. If multiple packages are specified, the names should be separated by spaces on the command line.

This option, as well as "2" and "3," displays a list of all available packages in the currently selected set. Each line describes the package name, whether the package is fully installed, not installed or partially installed, the size of the package (in 512 byte blocks), and a one line description of the package contents.

2. Remove

Prompts for one or more package names.

Deletes the correct files in the specified package. If multiple packages are specified the names should be separated by spaces on the command line.

Displays available packages (see option "1").

3. List files in a package

Lists all files in the specified package.

Prompts for one or more package names. Enter the name of the desired package(s).

Displays available packages (see option "1").

4. Install a single file

Extract the specified file from the distribution set.

Filename should be a full pathname relative to the root directory “/”.

5. Select a new set

Allows you to work from a different set than the current one.

6. Display current disk usage

Tells you your current disk usage.

7. Help

Prints a page of instructions to help you use *custom*.

Options

Three arguments are required for a completely non-interactive use of *custom*:

A set identifier

(**-o**, **-d**, or **-t**),

A command

(**-i**, **-r**, **-l**, or **-f**),

And either one or more package names, or a file name

If any information is missing from the command line, *custom* prompts for the missing data.

Only one of **-o**, **-d**, or **-t** may be specified. These stand for:

-o Operating System

-d Development System

-t Text Processing System

Only one of **-i**, **-r**, **-l**, or **-f** may be specified, followed by an argument of the appropriate type (one or more package names, or a file name). These options perform the following:

- i Install the specified package(s)
- r Remove the specified package(s)
- l List the files in the specified package(s).
- f Install the specified file.

The **-m** flag allows the media device to be specified. The default is `/dev/install` (which is always the 0 device, as in `/dev/fd0`). This is very useful if the system has a 5.25-inch drive on `/dev/fd0` and a 3.5-inch floppy on `/dev/fd1`, and it is necessary to install 3.5-inch media. For example:

```
custom -m /dev/rfd196ds9
```

this will override the default device and use the one supplied with the **-m** flag.

Files

```
/etc/base.perms  
/etc/soft.perms  
/etc/text.perms  
/etc/perms/*
```

See Also

`fixperm(ADM)`, `df(C)`, `du(C)`, `install(ADM)`

Notes

If you upgrade any part of your system, *custom* detects if you have a different release and prompts you to insert the floppy volume that updates the custom data files. Likewise, if you insert an invalid product or a volume out of order, you will be prompted to reinsert the correct volume.

Name

dial, uchat - Dials a modem.

Syntax

```
/usr/lib/uucp/dialX ttyname telno speed
/usr/lib/uucp/dialX -h ttyname speed
/usr/lib/uucp/uchat ttyname speed chat-script
```

Description

/usr/lib/uucp/dialX dials a modem attached to *ttyname*. (*X* is a dialer name, such as **HA1200**.) The **-h** option is used to hang up the modem.

uucico(ADM), **ct**(C), and **cu**(C) use **/usr/lib/uucp/dialX**. Four dialer programs are distributed. **dialHA12** is for the Hayes® Smartmodem 1200 and 1200B (and compatibles). **dialHA24** is for the Hayes® Smartmodem 2400 (and compatibles). **dialVA3450** is for the Racal-Vadic VA3450-Series dialers. **dialTBIT** is for the Telebit Trailblazer. Source for these is provided in their respective *.c* files.

uucico(ADM) invokes *dial*, with a *ttyname*, *telno* (phone number), and *speed*. *dial* attempts to dial the phone number on the specified line at the given speed. When using the **dialHA12** or **dialHA24** *speed* can be a range of baud rates. The range is specified with the form:

lowrate - *highrate*

where *lowrate* is the minimum acceptable connection baud rate and *highrate* is the maximum. The *dial* program returns the status of the attempt through the following dial return codes:

bit 0x80 = 1

The connection attempt failed.

bits 0x0f =

If bit 0x80 is a 1, then these bits are the dialer error code:

- | | |
|---|----------------------------------|
| 0 | general or unknown error code. |
| 1 | line is being used. |
| 2 | a signal has aborted the dialer. |
| 3 | dialer arguments are invalid. |

4	the phone number is invalid.
5	the baud rate is invalid or the dialer could not connect at the requested baud rate.
6	can't open the line.
7	ioctl error on the line.
8	timeout waiting for connection.
9	no dialtone was detected.
10	unused.
11	unused.
12	unused.
13	phone is busy.
14	no carrier is detected.
15	remote system did not answer.

Error codes 12-15 are used to indicate that the problem is at the remote end.

If bit 0x80 is a 0, then these bits are used to indicate the actual connection baud rate. If 0, the baud rate is the same as the baud rate used to dial the phone number or the highest baud rate if a range was specified. Otherwise, these four bits are the CBAUD bits in the **struct termio c_flag** and the **struct sgttyb sg_ispeed** and **sg_ospeed** tty ioctl structures.

You can copy and modify one of the files **/usr/lib/uucp/dialHA12.c** etc., to use a different modem. There is a makefile in **/usr/lib/uucp** which should be modified for the new dialer, and can be used to compile the new program.

If you create a *dial* program for another modem, send us the source. User generated *dial* programs will be considered for inclusion in future releases.

The **dial** program to be used on a particular line is specified in the fifth field of the entry for that line in **/usr/lib/uucp/Devices**. If there is no **dial** program of that name, then **uucico**, **ct**, and **cu** use a built-in dialer, together with the chat-script of that name in **/usr/lib/uucp/Dialers**.

dial -h is executed by **getty** when it is respawned on a line shared between dial-in and dial-out. If there is no **dial** program, then **getty** uses **/usr/lib/uucp/uuchat**, passing it the **&** chat-script from **/usr/lib/uucp/Dialers**.

Files

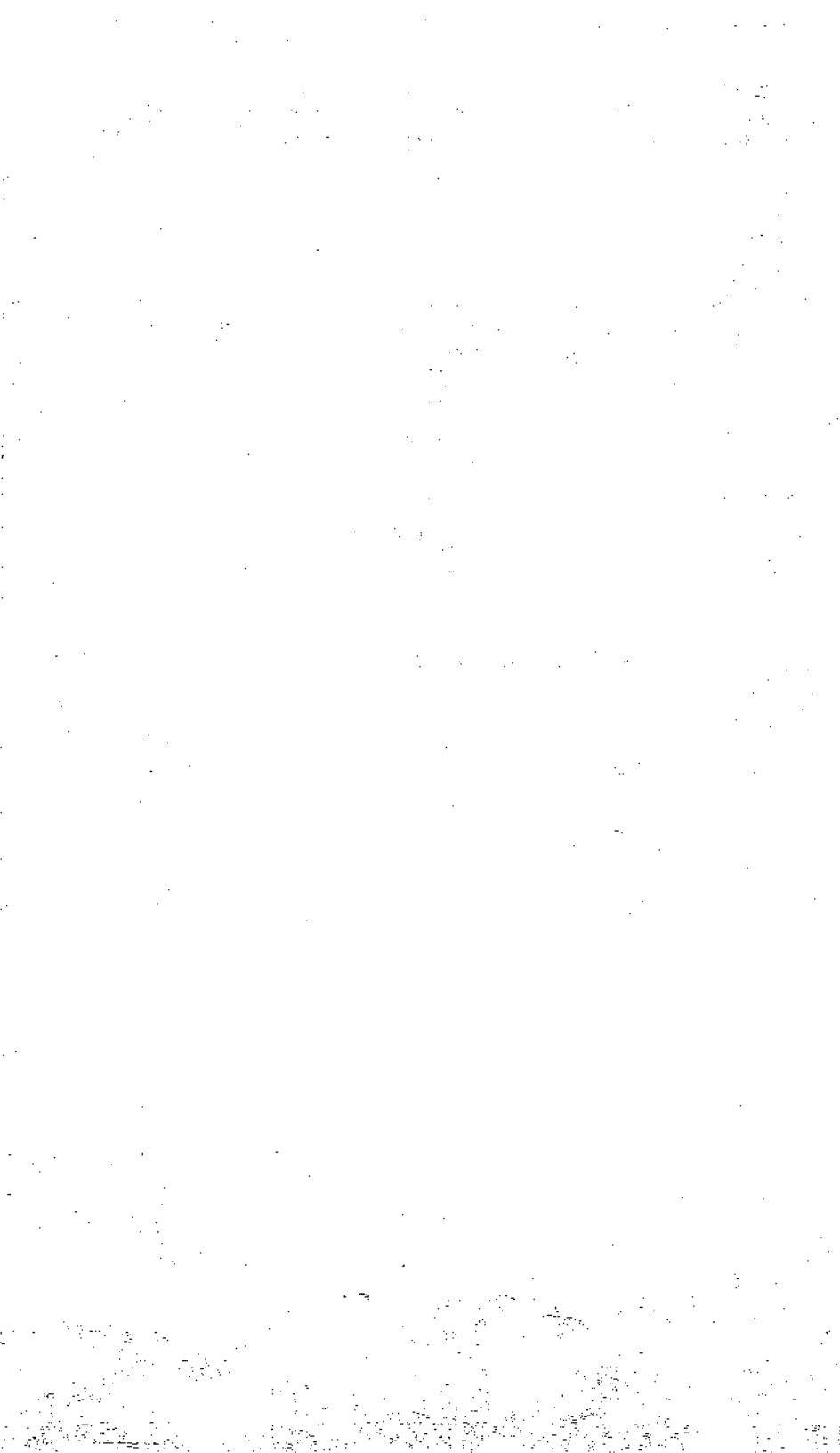
/usr/lib/uucp/Devices	
/usr/lib/uucp/dialVA3450	Racal Vadic 3450 dialer
/usr/lib/uucp/dialHA12	Hayes Smartmodem 1200/1200B dialer
/usr/lib/uucp/dialHA24	Hayes Smartmodem 2400 dialer
/usr/lib/uucp/makefile	Makefile to compile new dialer
/usr/lib/uucp/dialTBIT	Telebit Trailblazer dialer
/usr/lib/uucp/uuchat	

See Also

ct(C), cu(C), uucico(ADM), getty(M)

Notes

You must have the XENIX Development System installed in order to compile and install a new **dial** program.



Name

divvy - Disk dividing utility

Syntax

```
divvy -b block_device -c character_device [-v virtual_drive]
[-p physical_drive] [-i ] [-m ]
```

Description

divvy divides an *fdisk*(ADM) partition into a number of separate areas known as “divisions”. A division is identified by unique major and minor device numbers and can be used for a file system, swap area, or for isolating bad spots on the device.

With *divvy* you can:

- Divide a disk or *fdisk* partition into separate devices.
- Create new file systems.
- Change the device names of file systems.
- Change the size of file systems.
- Remove file systems.

Options

Options to *divvy* are:

- b *block_device*
Major device number of block interface.
- c *character_device*
Major device number of character interface.
- v *virtual_device*
For dividing a virtual drive.
- p *physical_drive*
For dividing one of several physical disks that share the same controller.
- i Disk being divided will contain a **root** file system on division 0.
- m Disk being divided should be made into a number of mountable file systems.

Usage

The device being divided must be a block device with a character interface. For example, to use *divvy* on a device with a block-interface major number 1 and character interface number of 1, enter:

```
divvy -b 1 -c 1
```

The **-v** option specifies which virtual drive to divide. The default is the active drive. Here, “virtual drive” is the same as an MS-DOS partition. Virtual drive numbers are determined with the *fdisk*(ADM) utility.

The **-p** option allows division of one of several physical disks sharing a controller. *divvy* defaults to the first physical device numbered “0.” To access a second physical disk, use the **-p 1** option.

The **-i** option specifies the device being divided will contain a **root** file system. With this option, device nodes are created relative to the new **root**, generally a hard disk, instead of the current **root**, often an installation floppy. A root filesystem and a recover area are created. *divvy* prompts for the size of the swap area. If the disk is large enough, then *divvy* prompts for a separate **/u** (user) filesystem. *divvy* also prompts for block-by-block control over the layout of the filesystem(s). If the root filesystem is large enough to require a scratch filesystem, (more than 40,000 blocks) then *divvy* will prompt for whether one should be created. *divvy* is invoked with the **-i** option during XENIX installation.

The **-m** option is used for initial installation on devices that will not be used as the root. It causes the user to be prompted for a number of file systems.

When *divvy* is invoked from the command line, you see a main menu:

n[ame]	Name or rename a division.
c[reate]	create a new file system on this division.
p[revent]	Prevent a new file system from being created on this division.
s[tart]	Start a division on a different block.
e[nd]	End a division on a different block.
r[estore]	Restore the original partition table.

Please enter your choice or ‘q’ to quit:

To choose a command, enter the first letter of the command, then press RETURN.

The *divvy* division table might look something like this:

Name	New File System?	#	First Block	Last Block
root	no, exists	0	0	13754
swap	no, exists	1	13755	15135
u	no, exists	2	15136	25135
	no	3	—	—
	no	4	—	—
	no	5	—	—
recover	no, exists	6	25136	25145
d1057all	no	7	0	25546

x blocks for divisions, *y* blocks reserved for the system

divvy also displays information about block allocation for system tables and bad tracks.

If you select option 'n', you can change the name of the device. *divvy* prompts you for the division number (from the *divvy* table displayed above), then for a new name.

Option 'c' causes a given division to become a new, empty file system when you exit from *divvy*. After using the 'c' option, you will see a 'yes' in the 'New File System?' column. If you use option 'p,' the 'yes' in the 'New File System?' column will change to a 'no', and the contents of the division will not change.

With the 's' or 'start' command, you can start a division on a different block number. With the 'e' or 'end' command, you can end a division on a different block number.

You can use these two commands to change the size of a partition. For example, if your disk is similar to the one in the sample *divvy* table above, and you want to make the **root** file system larger and the **swap** area smaller, do this:

Make the swap area smaller with the 's' command.

Use the 'e' command to make the **root** division bigger.

Note that if any of the divisions overlap, *divvy* will complain when you try to exit and put you back in the menus to correct the situation.

The 'r' or 'restore' command restores the original partition table. This is useful if you make a serious mistake and want to return to where you started.

When you exit from *divvy*, you are prompted whether you want to save any changes you made, or exit without saving the changes. At this time, you can also go back to the *divvy* menu, and may also have the option to reinstall the original, default partition table.

See Also

badtrk(ADM), fdisk(ADM), fsck(ADM), hd(M), mkdev(C), mkfs(C), mknod(C)

Notes

divvy requires kernel level support from the device driver. If *divvy* lists the size of a disk as “0” blocks, or displays the following error messages, the device may not support dividing:

cannot read division table

or:

cannot get drive parameters

These errors may also occur if the prerequisite programs *fdisk* and *badtrk* are not run correctly.

If you change the size of filesystems (such as */u*) after you have installed a XENIX filesystem, you will have to run *mkfs* on the filesystem and reinstall the files that are kept there. This is because the free list for that filesystem has changed. Be sure to backup the files in any filesystem you intend to change, using *backup*(C), *tar*(C), or *cpio*(C), before you run *divvy*. After XENIX is installed, the bounds of the **root** file system must not be changed.

During installation, if the filesystem on division 0 (generally root) becomes or remains large enough to require a scratch area during *fsck*, and one does not already exist, *divvy* prompts for whether one should be created. (The resulting filesystem, **/dev/scratch**, is used by *auto-boot* if it runs *fsck*. **/dev/scratch** should also be entered when *fsck* prompts for a scratch file name, provided that the filesystem being checked is not larger than the root filesystem.) If all disk divisions have been used up, *divvy* will not prompt for a scratch filesystem, even if the root filesystem is large enough to require one.

This utility uses BSIZE blocks. Refer to the *machine (HW)* manual page for the size of filesystem blocks.

Name

`dmesg` - Displays the system messages on the console.

Syntax

`dmesg [-]`

Description

The `dmesg` command displays all the system messages that have been generated since the last time the system was booted. If the option `—` is specified, it displays only those messages that have been generated since the last time the `dmesg` command was performed.

`dmesg` can be invoked periodically by placing instructions in the file `/usr/lib/crontab` . It can also be invoked automatically by `/etc/rc` whenever the system is booted. See “Notes”, below.

`dmesg` logs all error messages it prints in `/usr/adm/messages`. If `dmesg` is invoked automatically, the `messages` file continues to grow and can become very large. The system administrator should occasionally erase its contents.

Files

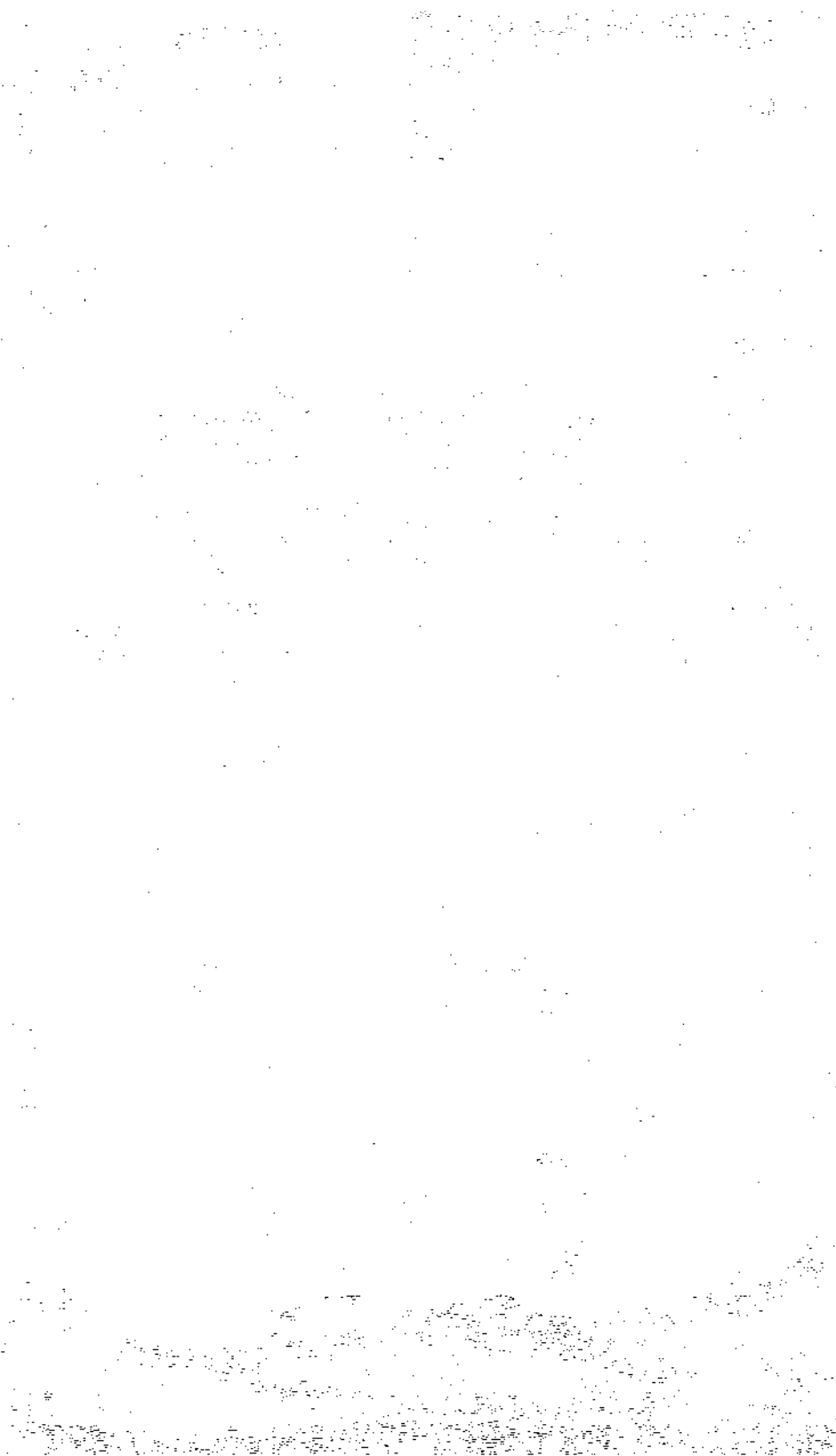
`/etc/dmesg`
`/usr/adm/messages`
`/usr/adm/msgbuf`

Notes

`dmesg` is included in this release for backwards compatibility only. The device `/dev/error` provides a more flexible means of logging error messages, and is recommended over `dmesg`. See `error(M)` for more information.

See Also

`cron(C)`, `error(M)`, `messages(M)`



Name

`dparam` - Displays/changes hard disk characteristics.

Syntax

```
dparam [ -w ]
dparam /dev/rhd[0|1]0 [characteristics]
```

Description

The *dparam* command displays or changes the hard disk characteristics currently in effect. These changes go into effect immediately and are also written to the master boot block for subsequent boots. If a non-standard hard disk is used, this utility must be called before accessing the drive.

-w Causes a copy of **/etc/masterboot** to be copied to disk to ensure that non-standard hard disks are supported for the specified drive. This call must precede a call to write non-standard disk parameters for the desired parameters to be saved correctly in the masterboot block.

When called without options or disk characteristics, *dparam* prints the current disk characteristics (on the standard output) for the specified hard disk. These values are printed in the same order as the argument list.

When writing characteristics for the specified hard disk, *dparam* changes the current disk controller status and updates the masterboot block. The argument ordering is critical and must be entered as specified below. All characteristics must be entered when writing disk characteristics, otherwise an error is returned. Hard disk characteristics (in respective order) are:

number of cylinders	total number of cylinders on the hard disk
number of heads	number of heads
write cylinder	hardware specific, consult your hardware manual
write precompensation cylinder	hardware specific, consult your hardware manual

ecc	number of bits of error correction on I/O transfers, consult your hardware manual
control	very hardware specific, consult your hardware manual
landing zone cylinder	where to park heads after shutting down the system
number of sectors per track	number of sectors per track on the hard disk

Examples

```
dparam -w
```

```
dparam /dev/rhd10
```

```
dparam /dev/rhd00 700 4 256 180 5 0 640 17
```

Notes

This utility changes the kernel's view of the hard disk parameters. It may be subject to restrictions imposed by the hardware configuration.

Name

`fdisk` - Maintain disk partitions.

Syntax

`fdisk` **[[-p] [-ad partition] [-c partition start size] [-f devicename]]**

Description

fdisk displays information about disk partitions. *fdisk* also creates and deletes disk partitions and changes the active partition. *fdisk* functionality is a superset of the MS-DOS command of the same name. *fdisk* is usually used interactively from a menu.

The hard disk has at most four partitions. Only one partition is active at any given time. It is possible to assign a different operating system to each partition. Once a partition is made active, the operating system resident in that partition boots automatically once the current operating system is halted.

To use XENIX, at least one partition must be assigned to XENIX.

The *fdisk* utility does not allocate the first track or the last cylinder on the hard disk when the "Use Entire Disk for XENIX" option is used. The first track on the hard disk is reserved for masterboot and the last cylinder is generally used when running hard disk diagnostics. You should not allocate the last cylinder if you plan to run diagnostics on your hard disk.

For example, if a disk has 2442 tracks, *fdisk* reports these as tracks 0-2441. If your hard disk has 4 heads, *fdisk* will assign (using the "Use Entire Disk for XENIX" option) tracks 1-2437. (Track 0 is reserved for masterboot.) The last cylinder (tracks 2438-2441) is not assigned with the "Use Entire Disk for XENIX" option.

Partitions are defined by a "partition table" at the end of the master boot block. The partition table provides the location and size of the partitions on the disk. The partition table also defines the active partition. Each partition can be assigned to XENIX, DOS, or some other operating system. Once a DOS partition is set up, DOS files and directories resident in the DOS partition may be accessed while running XENIX by means of the *dos(C)* commands. DOS may be booted without the DOS partition being active via the "boot:dos" command. See *boot(HW)*.

Arguments

-p, -a, -d, -c

These flags are used to invoke *fdisk* non-interactively:

-p	prints out the disk partition table.
-a <i>number</i>	activates the specified partition number.
-d <i>number</i>	deletes the specified partition number.
-c <i>number start size</i>	creates partition with specified start and size.

-f *name*

Open device *name* and read the partition table associated with that device's partition. The default is */dev/rhd00*.

Options

The *fdisk* command displays a prompt and a menu of five options. Updates to the disk are not made until you enter "q" from the main menu.

1. Display Partition Table.

This option displays a table of information about each partition on the hard disk. The PARTITION column gives the partition number. The STATUS column tells whether the partition is active (A) or inactive (I). TYPE tells whether the partition is XENIX, DOS, or "other". The option also displays the starting track, ending track and total number of tracks in each partition.

2. Use Entire Disk for XENIX.

fdisk creates one partition that includes all the tracks on the disk, except the first track and the last cylinder. This partition is assigned to XENIX and is designated the active partition.

3. Create XENIX Partition

This option allows the creation of a partition by altering the partition table. *fdisk* reports the number of tracks available for each partition and the number of tracks in use. *fdisk* prompts for the partition to create, the starting track and size in tracks. The change is written to the operating system and the hard disk when you enter "q" from the main menu.

4. Activate Partition

This option activates the specified partition. Only one partition may be active at a time. The change is not effective until you exit. The operating system residing in the newly activated partition boots once the current operating system is halted.

5. Delete Partition

This option requests which partition you wish to delete. *fdisk* reports the new available amount of disk space in tracks. The change is not effective until you exit.

Exit the *fdisk* program by typing a 'q' at the main *fdisk* menu. Your changes are now written to the operating system and the hard disk.

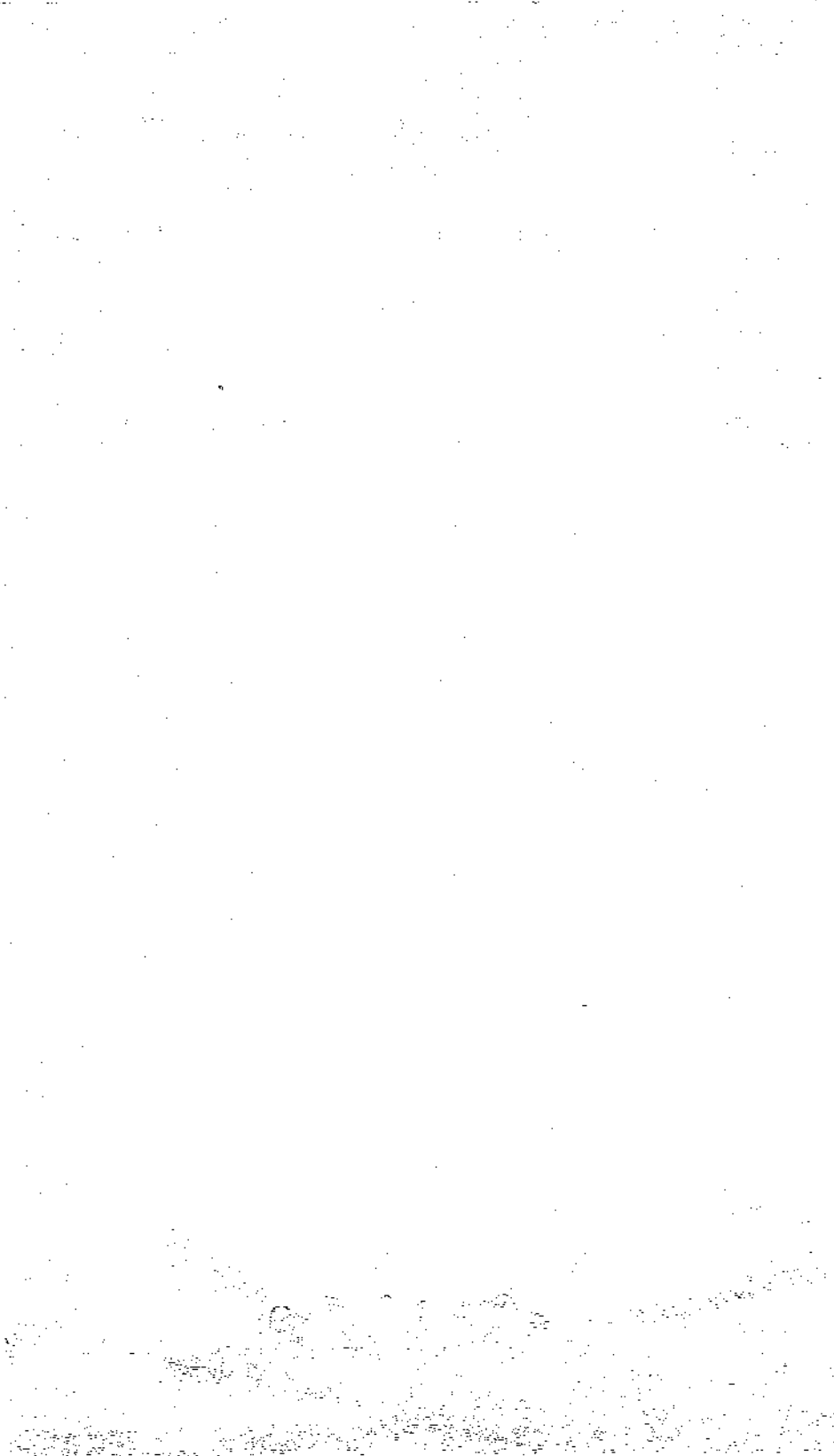
Notes

The minimum recommended size for a XENIX partition is 5 megabytes.

Since *fdisk* is intended for use with DOS, it may not work with all operating system combinations.

See also

dos(C), hd(HW).



Name

fdswap - swaps default boot floppy drive.

Syntax

fdswap [on|off]

Description

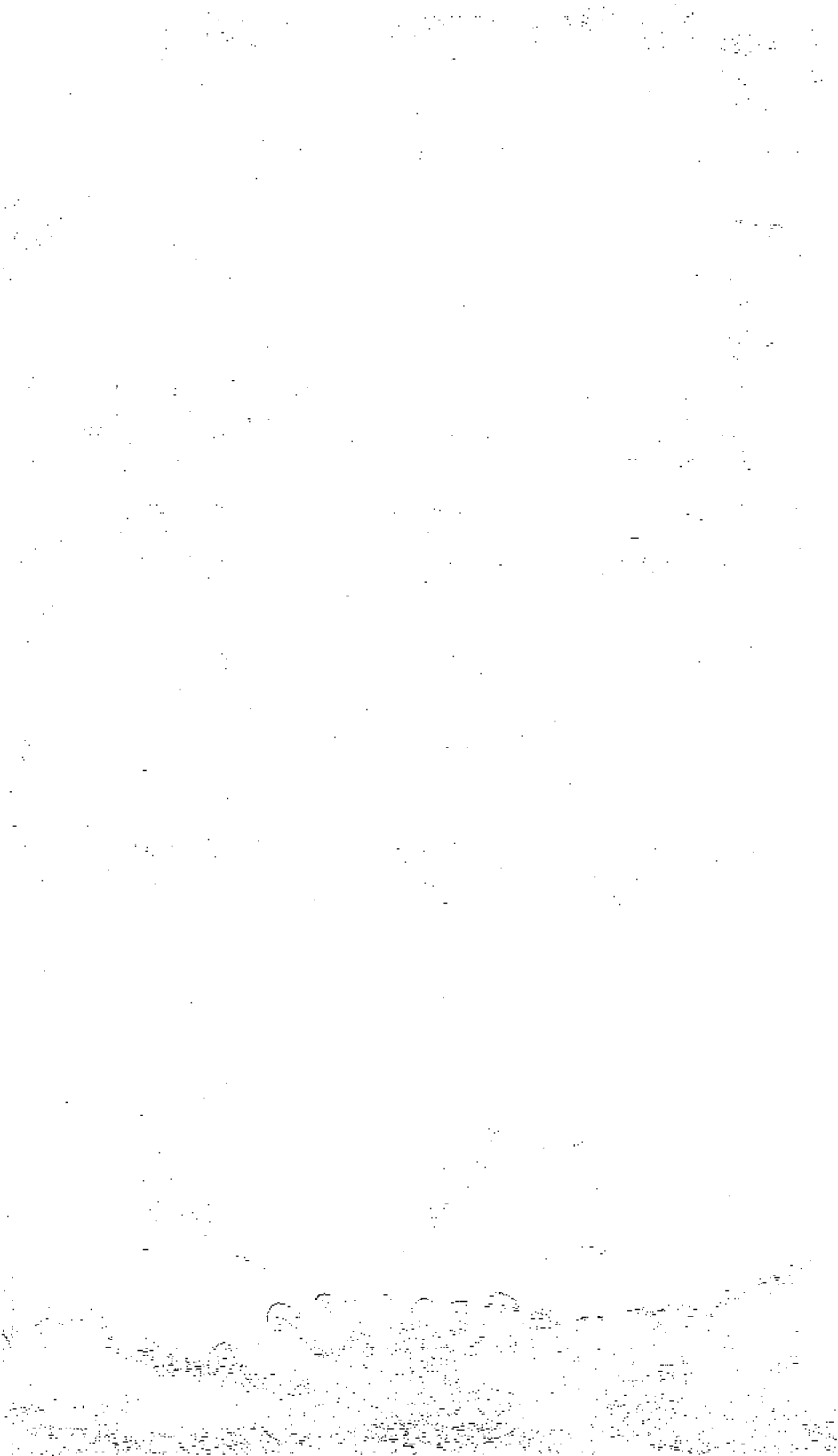
fdswap tells the CMOS to swap the default floppy drive used to read boot information at boot time. For example, if your computer defaults to read boot information on drive A, **fdswap on** changes the default drive to drive B.

fdswap with no arguments reports the current **fdswap** state, on or off. **fdswap off** switches the drive setting back to the default configuration. Changing the drives take effect on the next boot of the system.

Notes

<p>This utility is not applicable to all hardware/software configurations and may not be included in your distribution.</p>

Support for this functionality is only available on a small number of machines. The ROMs must recognize and interpret the CMOS flag that specifies that the floppy drives are swapped.



Name

fixperm - Correct or initialize file permissions and ownership.

Syntax

fixperm [**-c**fgilnsvwDS [**-d** package]] **specfile**

Description

For each line in the specification file **specfile**, *fixperm* makes the listed pathname conform to a specification. *fixperm* is typically used to configure a XENIX system upon installation. Non-superusers can only use *fixperm* with the **-n**, **-f**, **-D** or **-l** flags. To use any other flags, you must be superuser.

The specification file has the following format: Each non-blank line consists of either a comment or an item specification. A comment is any text from a pound sign “#” up to the end of the line. There is one item specification per line. User and group id numbers must be specified at the top of the specification file for each user and group mentioned in the file. The syntax for the definition section is simple: the first field indicates the type of id (either *uid* or *gid*), the second contains the name reference for the id, and the third is the corresponding numeric id. Example:

```
uid    root    0
```

An item specification consists of a package specifier, a permission specification, owner and group specifications, the number of links on the file, the file name, and an optional volume number.

The package specifier is an arbitrary string which is the name of a package within a distribution set. A package is a set of files.

After the package specifier is a permission specification. The permission specification consists of a file type, followed by a numeric permission specification. The item specification is one of the following characters:

- x Executable.
- a Archive.
- e Empty file (create if **-c** option given).
- b Block device.

- c Character device.
- d Directory.
- f Text file.
- p Named pipe.

If the item specification is used as an upper-case letter, then the file associated with it is optional, and *fixperm* will not return an error message if it does not exist.

The numeric permission conforms to the scheme described in *chmod(C)*. The owner and group permissions are in the third column separated by a slash: e.g.,: "bin/bin". The fourth column indicates the number of links. If there are links to the file, the next line contains the linked filename with no other information. The fifth column is a pathname. The pathname must be relative, i.e., not preceded by a slash "/". The sixth column is only used for special files, giving the major and minor device numbers, or volume numbers.

Options

The following options are available from the command line:

- c Create empty files and missing directories. Also or creates (or modifies) device files.
- g Instructs *fixperm* to list devices as specified in the permlist (similar to the -f flag, which lists files on standard output). No changes are made as a result of this flag.
- d *package*
Process input lines beginning with given package specifier string (see above). For instance, -dBASE processes only items specified as belonging to the Basic utilities set. The default action is to process all lines.
- u *package*
Like -u, but processes items that are not part of the given package.
- f List files only on standard output. Does not modify target files.
- i Check only if the selected packages are installed. Return values are:
 - 0: package completely installed
 - 4: package not installed
 - 5: package partially installed

- l List files and directories on standard output. Does not modify target files.
- n Report errors only. Does not modify target files.
- D List directories only on standard output. Does not modify target files.
- v Verbose, in particular, issues a complaint if executable files are word swapped, not fixed stack, not separate I and D, or not stripped.
- s Modify special device files in addition to the rest of the permlist.
- w Lists where (what volume) the specified files or directories are located.
- S Issues a complaint if files are not in x.out format.

The following two lines make a distribution and invoke *tar*(C) to archive only the files in *perms.inst* on */dev/sample*:

```
/etc/fixperm -f /etc/perms/inst > list
tar cfF /dev/sample list
```

This example reports *BASE* package errors:

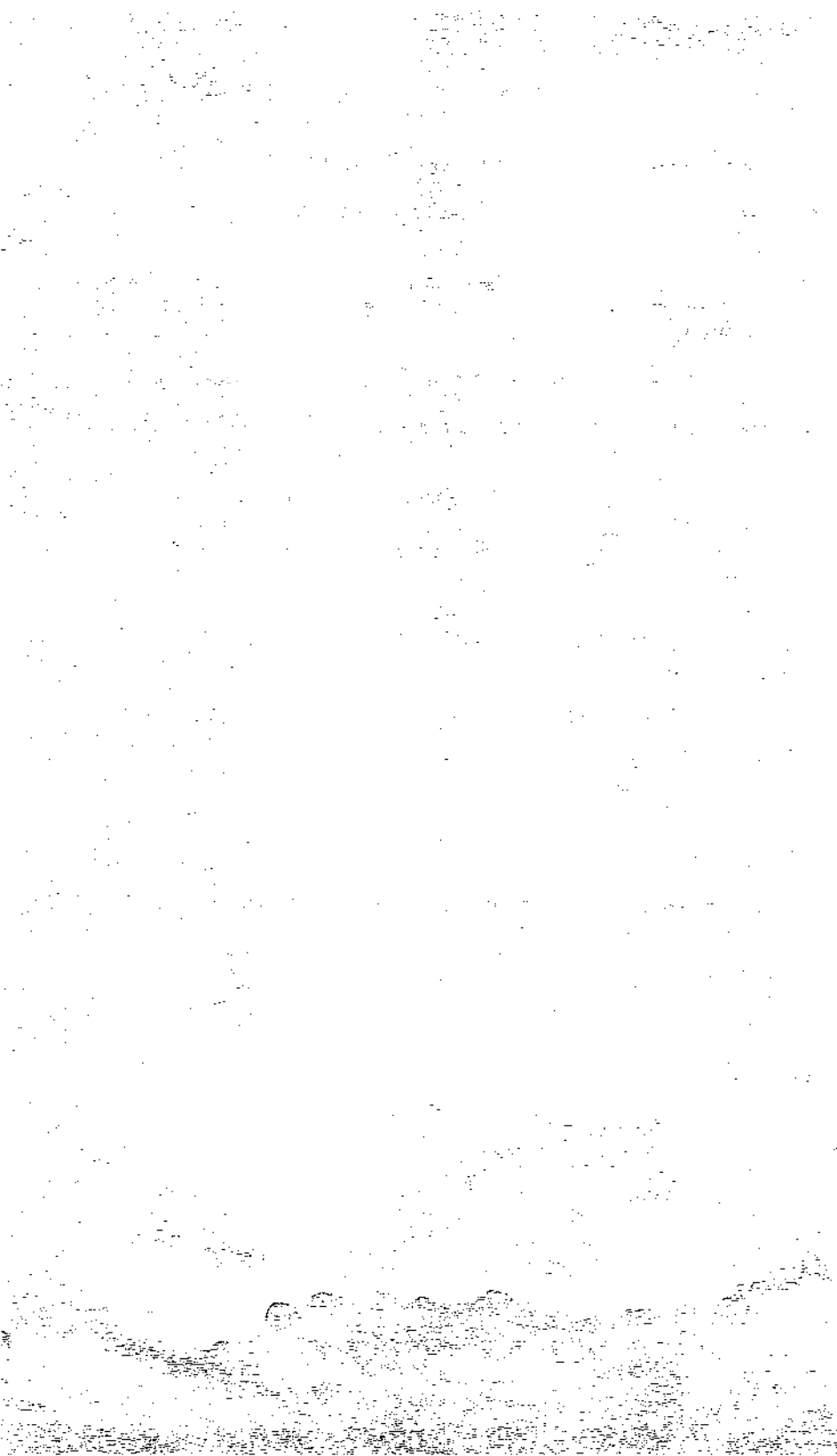
```
/etc/fixperm -nd BASE
```

Notes

Usually *fixperm* is only run by a shell script at installation.

See Also

custom (ADM)



Name

fsave - Interactive, error-checking filesystem backup

Synopsis

fsave filesystem [dumpinfo] [mediainfo] [sitename]

Description

fsave is used by *fsphoto*(ADM) to provide a semi-automated interface to *backup*(C) for backing-up XENIX filesystems. Human intervention is required to mount and dismount tapes or floppies at the appropriate times, but is kept to a minimum to reduce the potential for error.

The operator is prompted each time some action is required, such as mounting or unmounting a tape or floppy. These prompts, and their possible selections, are described below.

For all prompts, an answer of **h**, **H**, or **?** will display a short summary of the possible answers.

Filesystem dump (backup)

The following prompt displays the defaults (gleaned from the *schedule* database file) and presents options to alter them:

Level *dumplevel* dump of filesystem *filesystem*, *date*
media size: *size* feet [or Kb]
media drive: *drive*

This *media* will be saved for *howlong*, and is *howvital*.

M)ounted volume, P)ostpone, C)heck or F)ormat volumes, R)etension or H)elp:

The values displayed dictate the following instructions: *filesystem* is to be backed-up using *size*-foot long magtapes (or *size*-kilobyte big floppies) mounted on drive *drive*. The *media* will be saved for *howlong* ("1 year," "2 months," etc.), and being a level *dumplevel* dump, is *howvital* ("critical," "precautionary," etc.).

The menu options are:

m A volume of the asked for *size* has been mounted (write-enabled), so begin the dump.

mnewsiz Insufficient volumes of the originally asked for *size* are available, so a *newsiz* big volume has been mounted instead. If the dump extends across more than one volume, each volume must be of the same *size*.

- p** Postpone this backup until later (*fsphoto* will automatically retry this *filesystem* next time it is run).
- c** Recheck the volumes used to backup *filesystem* for errors. This answer is useful when a dump mysteriously fails and *fsave* is starting over from the beginning, but the operator doesn't believe there really is a problem (for example, the tape drive was accidentally left offline or the floppy door was left open), and wants to check the volumes again.
- f** Format the currently mounted volume (useful mainly for floppies).
- r** Retension cartridge tape using */usr/bin/tape*.

If multiple volumes are required, *backup* will pause for the next volume to be mounted. Be certain to keep track of the volume order.

Format check

The format of "critical" volumes are checked using *dumpdir(C)*:

Check *vital* volumes for format errors
 M)ounted first volume, S)kip format check, or H)elp:

The menu options are:

- m** The first volume has been (or still is) mounted, and *dumpdir* can now check the volume format.
- s** Skip checking the volume format, and continue on to the read error check (below).

The format is not always checked, but when it is, the first volume written must be mounted.

Read error check

All volumes are read using *restor(C)*, which checks for errors during reading. If an error occurs, the dump is declared unsuccessful and is retried from the beginning.

Check *vital* volumes for read errors
 M)ounted *which* volume, E)rror on previous volume, D)one, S)kip checks, or H)elp:

The menu options are:

- m** The *which* ("first" or "next") volume has been mounted on the drive and is ready to be checked for read errors.

- e An error occurred on the last volume checked, and the dump should be retried.
- d All volumes have been checked and no errors occurred, so the filesystem has been successfully backed-up; This backup is done.
- s Don't bother (skip) checking the rest of the volumes for read errors.

Every volume should be checked for read errors; *restor* requires the volumes to be checked in first-to-last order. Volumes that produce read errors should be marked "suspect," discarded and the dump run once again.

After the backup has been successfully performed, instructions are given on how to label the volumes.

Arguments

fsave is normally run by *fsphoto*, which passes all the proper arguments based on the *schedule* (ADM) database.

filesystem

The filesystem to be backed-up.

dumpinfo

A set of blank-separated strings that give some optional information about this backup:

dumplevel size savetime importance marker

Each of these component strings may be quoted and can thus contain spaces.

dumplevel The level of the dump to be performed. This is a single digit from 0 to 9 (passed to *dump*), or the letter x (which means no dump is to be done). The default is to perform a level 0 dump.

size The size of the media volumes that should be used. This should be in feet for tapes and kilobytes for floppies. A *size* of - means to use the first size listed in *mediainfo*. This is the default.

savetime How long this backup is to be saved (for example, "3 months"). Default is "1 year."

importance

How important is this backup? (For example, "critical" or "precautionary.") Those which are "critical" have their format checked by *dumpdir*. Default is

“important.”

marker Either “none” (the default) or an additional label to place on each volume (for example, “a pink sticker”).

A typical *dumpinfo* might look like:

```
9 1200 "2 weeks" useful "a blue X"
```

which specifies that a level 9 dump is to be done on a 1200 foot tape (or 1200 kilobyte floppy) which will be saved for 2 weeks and is to be marked with a blue cross (in addition to a more descriptive label). This backup is merely considered “useful” and thus will not be checked by *dumpdir*.

mediainfo

A set of blank-separated strings that give some optional information about this the media to be used:

```
drive d density sizes ... [format]
drive k sizes ... [format]
```

drive The name of backup device to use. The default is */dev/rmt0*.

k sizes... If *k* is specified, *drive* is assumed to be a floppy, and the list of *sizes* which follow define the allowable capacities of the floppies that can be used (in kilobytes).

d density sizes... Otherwise, *d* must be specified. In this case, *drive* is assumed to be a magtape at *density* BPI, in one of the possible *sizes* (in feet).

format The XENIX command used to format the tape or floppy so described.

A *mediainfo* describing 9-track magtape would be:

```
media /dev/rmt0 d 1600 2400 1200 600
media /dev/rmt2 d 800 1400 1200 600
```

which specifies that */dev/rmt0* is a 1600 BPI magtape capable of handling 2400, 1200, and 600 foot reels, and that */dev/rmt2* is the 800 BPI device.

A floppy might be described with:

```
media /dev/fd0 k 1024 format /dev/fd0
```


which describes device */dev/fd0* as a megabyte (1024 kilobytes) floppy formatted by the command:

```
format /dev/fd0
```

sitename

Where this backup was made (for example, the name of the company or which building). Note that the *uucp(C)* nodename from */etc/systemid* is automatically placed on the volume labels.

Only the super-user can execute the *fsave* command.

Files

/etc/systemid

Name of this machine.

/etc/ddate

Dump-maintained record of last time each filesystem was backed-up.

/dev/tty

Always-existent character-special device.

See Also

fsphoto(ADM), *schedule(ADM)*, *backup(C)*, *dumpdir(C)*, *restor(C)*, *basename(C)*,

Diagnostics

A successful backup exits successfully (0), but errors generate a complaint and an exit status of 1. *fsave* complains about illegal or incorrect arguments, and exits with a status of 2.

If the backup of *filesystem* is postponed, *fsave* exits with a status of 3.



Name

fsck - Checks and repairs filesystems.

Syntax

`/bin/fsck [options] [filesystem] ...`

Description

fsck audits and interactively repairs inconsistent conditions for XENIX System V filesystems. If the filesystem is consistent, the the number of files, number of blocks used, and number of blocks free are reported. If the filesystem is inconsistent, the operator is prompted for concurrence before each correction is attempted. It should be noted that most corrective actions result in some loss of data. The amount and severity of the loss may be determined from the diagnostic output. (An experienced operator can resolve discrepancies manually using *fsdb*(ADM), the filesystem debugger.) The default action for each consistency correction is to wait for the operator to respond "yes" or "no". If the operator does not have write permission *fsck* defaults to the action of the **-n** option.

The following flags are interpreted by *fsck*:

- y** Assumes a yes response to all questions asked by *fsck*.
- n** Assumes a no response to all questions asked by *fsck*; do not open the filesystem for writing.
- sb:c** Ignores the actual free list and (unconditionally) reconstructs a new one by rewriting the super-block of the filesystem. The filesystem *must* be unmounted while this is done.

The **-sb:c** option allows for creating an optimal free-list organization. The following forms are supported:

```
-s
-sBlocks-per-cylinder:Blocks-to-skip (filesystem interleave)
  (for anything else)
```

If *b:c* is not given, then the values used when the filesystem was created are used. If these values were not specified, then a reasonable default value is used.

- S** Conditionally reconstructs the free list. This option is like **-sb:c** above except that the free list is rebuilt only if there are no discrepancies discovered in the filesystem. Using **-S** forces a "no" response to all questions asked by *fsck*. This option is useful for forcing free list reorganization on uncontaminated

filesystems.

- t If *fsck* cannot obtain enough memory to keep its tables, it uses a scratch file. If the **-t** option is specified, the file named in the next argument is used as the scratch file, if needed. Make certain you leave a space between the **-t** and the filename, or *fsck* will use the entire filesystem as a scratch file and erase the entire disk. If you created a scratch filesystem during installation then you can use `/dev/scratch` as the filename, provided that the filesystem being checked is no larger than the **root** filesystem. Without the **-t** flag, *fsck* prompts the operator for the name of the scratch file. The file chosen should not be on the filesystem being checked, and if it is not a special file or did not already exist, it is removed when *fsck* completes. If the system has a large hard disk there may not be enough space on another filesystem for the scratch file. In such cases, if the system has a floppy drive, use a blank, formatted floppy in the floppy drive with (for example) `/dev/fd0` specified as the scratch file.
- q Quiet *fsck*. Do not print size-check messages in Phase 1. Unreferenced **fifo5** files will selectively be removed. If *fsck* requires it, counts in the superblock will be automatically fixed and the free list salvaged.
- D Directories are checked for bad blocks. Useful after system crashes.
- f Fast check. Check block and sizes (Phase 1) and check the free list (Phase 5). The free list will be reconstructed (Phase 6) if it is necessary.
- rr Recovers the root filesystem. The required *filesystem* argument must refer to the root filesystem, and preferably to the block device (normally `/dev/root`). This switch implies **-y** and overrides **-n**. If any modifications to the filesystem are required, the system will be automatically shutdown to ensure the integrity of the filesystem.
- c Causes any supported filesystem to be converted to the type of the current filesystem. The user is prompted to verify the request for each filesystem that requires conversion unless the **-y** option is specified. It is recommended that every filesystem be checked with this option *while unmounted* if it is to be used with the current version of XENIX. To update the active root filesystem, it should be checked with:

```
fsck -c -rr /dev/root
```

If no *filesystems* are specified, *fsck* reads a list of default filesystems from the file `/etc/checklist`.

Inconsistencies checked are as follows:

- Blocks claimed by more than one inode or the free list
- Blocks claimed by an inode or the free list outside the range of the filesystem
- Incorrect link counts
- Size checks:
 - Incorrect number of blocks
 - Directory size not 16-byte aligned
- Bad inode format
- Blocks not accounted for anywhere
- Directory checks:
 - File pointing to unallocated inode
 - Inode number out of range
- Super block checks:
 - More than 65536 inodes
 - More blocks for inodes than there are in the filesystem
- Bad free block list format
- Total free block or free inode count incorrect

Orphaned files and directories (allocated but unreferenced) are, with the operator's concurrence, reconnected by placing them in the **lost+found** directory. The name assigned is the inode number. The only restriction is that the directory **lost+found** must preexist in the root of the filesystem being checked and must have empty slots in which entries can be made. This is accomplished by making **lost+found**, copying a number of files to the directory, and then removing them (before *fsck* is executed).

Files

<code>/etc/checklist</code>	Contains default list of filesystems to check
<code>/etc/default/boot</code>	Automatic boot control

See Also

autoboot(ADM), fsdb(ADM), checklist(F), filesystem(F), init(M)

Notes

fsck will not run on a mounted non-raw filesystem unless the filesystem is the root filesystem or unless the **-n** option is specified and no writing out of the filesystem will take place. If any such attempt is made, a warning is displayed and no further processing of the filesystem is done for the specified device.

Although checking a raw device is almost always faster, there is no way to tell if the filesystem is mounted. And cleaning a mounted filesystem will almost certainly result in an inconsistent superblock.

Warning

File systems created under XENIX-86 version 3.0 are not supported under XENIX System V because the word ordering in type *long* variables has changed. *fsck* is capable of auditing and repairing XENIX version 3.0 file systems if the word ordering is correct.

For the root filesystem, “*fsck -rr /dev/root*” should be run. For all other filesystems, “*fsck /dev/??*” on the *unmounted* block device should be used.

Diagnostics

Initialization Phase

Command syntax is checked. Before the filesystem check can be performed, **fsck** sets up certain tables and opens some files. The **fsck** terminates on initialization errors.

General Errors

Three error messages may appear in any phase. While they seem to offer the option to continue, it is generally best to regard them as fatal, end the run, and investigate what may have caused the problem.

CAN NOT SEEK: BLK B (CONTINUE?)

The request to move to a specified block number *B* in the filesystem failed. The occurrence of this error condition indicates a serious problem (probably a hardware failure) that may require additional help.

CAN NOT READ: BLK B (CONTINUE?)

The request for reading a specified block number *B* in the filesystem failed. The occurrence of this error condition indicates a serious problem (probably a hardware failure) that may require additional help.

CAN NOT WRITE: BLK B (CONTINUE?)

The request for writing a specified block number *B* in the filesystem failed. The disk may be write-protected.

Meaning of Yes/No Responses

Prompt	n(no)	y(yes)
CONTINUE?	Terminates program. (This is the recommended response.)	Attempts to continue to run filesystem check. Often, however, the problem persists. The error condition does not allow a complete check of the filesystem. A second run of <i>fsck</i> should be made to recheck this filesystem.

Phase 1: Check Blocks and Sizes

This phase checks the inode list.

Meaning of Yes/No Responses—Phase 1

Prompt	n(no)	y(yes)
CONTINUE?	Terminates the program. (Recommended response.)	Continues with the program. This error condition means that a complete check of the filesystem is not possible. A second run of <i>fsck</i> should be made to recheck this filesystem.
CLEAR?	Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.	Deallocates i-node <i>I</i> by zeroing its contents. This may invoke the UNALLOCATED error condition in Phase 2 for each directory entry pointing to this i-node.

Phase 1 Error Messages

UNKNOWN FILE TYPE I=I (CLEAR?)

The mode word of the i-node *I* suggests that the i-node is not

a pipe, special character i-node, regular i-node, or directory i-node.

LINK COUNT TABLE OVERFLOW (CONTINUE?)

An internal table for *fsck* containing allocated i-nodes with a link count of zero has no more room.

B BAD I=I

I-node *I* contains block number *B* with a number lower than the number of the first data block in the filesystem or greater than the number of the last block in the filesystem. This error condition may invoke the EXCESSIVE BAD BLKS error condition in Phase 1 if i-node *I* has too many block numbers outside the filesystem range. This error condition invokes the BAD/DUP error condition in Phase 2 and Phase 4.

EXCESSIVE BAD BLOCKS I=I (CONTINUE?)

There is more than a tolerable number (usually 10) of blocks with a number lower than the number of the first data block in the filesystem or greater than the number of the last block in the filesystem associated with i-node *I*.

B DUP I=I

I-node *I* contains block number *B*, which is already claimed by another i-node. This error condition may invoke the EXCESSIVE DUP BLKS error condition in Phase 1 if i-node *I* has too many block numbers claimed by other i-nodes. This error condition invokes Phase 1B and the BAD/DUP error condition in Phase 2 and Phase 4

EXCESSIVE DUP BLKS I=I (CONTINUE?)

There is more than a tolerable number (usually 10) of blocks claimed by other i-nodes.

DUP TABLE OVERFLOW (CONTINUE?)

An internal table in *fsck* containing duplicate block numbers has no more room.

POSSIBLE FILE SIZE ERROR I=I

The i-node *I* size does not match the actual number of blocks used by the i-node. This is only a warning. If the *-q* option is used, this message is not printed.

DIRECTORY MISALIGNED I=I

The size of a directory i-node is not a multiple of 16. This is only a warning. If the *-q* option is used, this message is not printed.

PARTIALLY ALLOCATED INODE I=I (CLEAR?)

I-node *I* is neither allocated nor unallocated.

Phase 1B: Rescan for More DUPS

When a duplicate block is found in the filesystem, the filesystem is rescanned to find the i-node that previously claimed that block. When the duplicate block is found, the following information message is printed:

B DUP I=I

I-node *I* contains block number *B*, which is already claimed by another i-node. This error condition invokes the BAD/DUP error condition in Phase 2. I-nodes with overlapping blocks may be determined by examining this error condition and the DUP error condition in Phase 1.

Phase 2: Check Path Names

This phase removes directory entries pointing to bad inodes found in Phase 1 and phase 1B.

Meaning of Yes/No Responses—Phase 2

Prompt	n(no)	y(yes)
FIX?	Terminates the program since <i>fsck</i> will be unable to continue.	In Phase 2, a y(yes) response to the FIX? prompt says: Change the root i-node type to "directory." If the root i-node data blocks are not directory blocks, a very large number of error conditions are produced.

(Continued)

Prompt	n(no)	y(yes)
CONTINUE?	Terminates the program.	Ignores DUPS/BAD error condition in root i-node and attempt to continue to run the filesystem check. If root i-node is not correct, then this may result in a large number of other error conditions.
REMOVE?	Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.	Removes duplicate or unallocated blocks.

Phase 2 Error Messages

ROOT INODE UNALLOCATED. TERMINATING

The root i-node (always i-node number 2) has no allocate mode bits. The occurrence of this error condition indicates a serious problem. The program stops.

ROOT INODE NOT DIRECTORY (FIX?)

The root i-node (usually i-node number 2) is not directory i-node type.

DUPS/BAD IN ROOT INODE (CONTINUE?)

Phase 1 or Phase 1B found duplicate blocks or bad blocks in the root i-node (usually i-node number 2) for the filesystem.

I OUT OF RANGE I=I NAME=F (REMOVE?)

A directory entry *F* has an i-node number *I* that is greater than the end of the i-node list.

UNALLOCATED I=I OWNER=O MODE=M SIZE=S MTIME=T NAME=F (REMOVE?)

A directory entry *F* has an i-node *I* without allocate mode bits. The owner *O*, mode *M*, size *S*, modify time *T*, and filename *F* are printed. If the filesystem is not mounted and the *-n* option was not specified, the entry is removed automatically if the i-node it points to is character size 0.

DUP/BAD I=I OWNER=O MODE=M SIZE=S MTIME=T

DIR=F (REMOVE?)

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with directory entry *F*, directory i-node *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

DUP/BAD I=I OWNER=O MODE=M SIZE=S MTIME=T FILE=F (REMOVE?)

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with file entry *F*, i-node *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and filename *F* are printed.

BAD BLK B IN DIR I=I OWNER=O MODE=M SIZE=S MTIME=T

This message only occurs when the *-D* option is used. A bad block was found in DIR i-node *I*. Error conditions looked for in directory blocks are nonzero padded entries, inconsistent "." and ".." entries, and embedded slashes in the name field. This error message means that the user should at a later time either remove the directory i-node if the entire block looks bad or change (or remove) those directory entries that look bad.

Phase 3: Check Connectivity

This phase is concerned with the directory connectivity seen in Phase 2.

Meaning of Yes/No Responses—Phase 3

Prompt	n(no)	y(yes)
RECONNECT?	<p>Ignores the error condition.</p> <p>This invokes the UNREF error condition in Phase 4.</p> <p>A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p>Reconnects directory i-node <i>I</i> to the filesystem in directory for lost files (usually <i>lost+found</i>).</p> <p>This may invoke a <i>lost+found</i> error condition if there are problems connecting directory i-node <i>I</i> to <i>lost+found</i>.</p> <p>This invokes CONNECTED information message if link was successful.</p>

Phase 3 Error Messages

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T
(RECONNECT?)

The directory i-node *I* was not connected to a directory entry when the filesystem was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of directory i-node *I* are printed. The *fsck* program forces the reconnection of a nonempty directory.

SORRY. NO lost+found DIRECTORY

There is no *lost+found* directory in the root directory of the filesystem; *fsck* ignores the request to link a directory in *lost+found*. This invokes the UNREF error condition in Phase 4. Possible problem with access modes of *lost+found*.

SORRY. NO SPACE IN lost+found DIRECTORY

There is no space to add another entry to the *lost+found* directory in the root directory of the filesystem; *fsck* ignores the request to link a directory in *lost+found*. This invokes the UNREF error condition in Phase 4. Clean out unnecessary entries in *lost+found* or make *lost+found* larger (see Procedure 5.2).

DIR I=I1 CONNECTED. PARENT WAS I=I2

This is an advisory message indicating a directory i-node *I1* was successfully connected to the *lost+found* directory. The parent i-node *I2* of the directory i-node *I1* is replaced by the i-node number of the *lost+found* directory.

Phase 4: Check Reference Counts

This phase checks the link count information seen in Phases 2 and 3.

Meaning of Yes/No Responses—Phase 4

Prompt	n(no)	y(yes)
RECONNECT?	<p> Ignores this error condition. This invokes a CLEAR error condition later in Phase 4.</p>	<p> Reconnect i-node <i>I</i> to filesystem in the directory for lost files (usually <i>lost+found</i>). This can cause a <i>lost+found</i> error condition in this phase if there are problems connecting i-node <i>I</i> to <i>lost+found</i>.</p>
CLEAR?	<p> Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p> Deallocates the i-node by zeroing its contents.</p>
ADJUST?	<p> Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p> Replaces link count of file i-node <i>I</i> with <i>Y</i>.</p>
FIX?	<p> Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p> Replaces count in super-block by actual count.</p>

Phase 4 Error Messages

UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T
(RECONNECT?)

I-node *I* was not connected to a directory entry when the filesystem was traversed. The owner *O*, mode *M*, size *S*, and

modify time *T* of i-node *I* are printed. If the *-n* option is omitted and the filesystem is not mounted, empty files are cleared automatically. Nonempty files are not cleared.

SORRY. NO lost+found DIRECTORY

There is no *lost+found* directory in the root directory of the filesystem; *fsck* ignores the request to link a file in *lost+found*. This invokes the CLEAR error condition later in Phase 4. Possible problem with access modes of *lost+found*.

SORRY. NO SPACE IN lost+found DIRECTORY

There is no space to add another entry to the *lost+found* directory in the root directory of the filesystem; *fsck* ignores the request to link a file in *lost+found*. This invokes the CLEAR error condition later in Phase 4. Check size and contents of *lost+found*.

(CLEAR)

The i-node mentioned in the immediately previous UNREF error condition cannot be reconnected.

LINK COUNT FILE I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST?)

The link count for i-node *I*, which is a file, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* are printed.

LINK COUNT DIR I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X SHOULD BE Y (ADJUST?)

The link count for i-node *I*, which is a directory, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* of directory i-node *I* are printed.

LINK COUNT F I=I OWNER=O MODE=M SIZE=S MTIME=T. COUNT=X SHOULD BE Y (ADJUST?)

The link count for *F* i-node *I* is *X* but should be *Y*. The filename *F*, owner *O*, mode *M*, size *S*, and modify time *T* are printed.

UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR?)

I-node *I*, which is a file, was not connected to a directory entry when the filesystem was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of i-node *I* are printed. If the *-n* option is omitted and the filesystem is not mounted, empty files are cleared automatically. Nonempty directories are not cleared.

UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T
(CLEAR?)

I-node *I*, which is a directory, was not connected to a directory entry when the filesystem was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of i-node *I* are printed. If the *-n* option is omitted and the filesystem is not mounted, empty directories are cleared automatically. Nonempty directories are not cleared.

BAD/DUP FILE I=I OWNER=O MODE=M SIZE=S MTIME=T
(CLEAR?)

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with file i-node *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of i-node *I* are printed.

BAD/DUP DIR I=I OWNER=O MODE=M SIZE=S MTIME=T
(CLEAR?)

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with directory i-node *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of i-node *I* are printed.

FREE INODE COUNT WRONG IN SUPERBLK (FIX?)

The actual count of the free i-nodes does not match the count in the super-block of the filesystem. If the *-q* option is specified, the count will be fixed automatically in the super-block.

Phase 5: Check Free List

This phase checks the free-block list.

Meaning of Yes/No Responses—Phase 5

Prompt	n(no)	y(yes)
CONTINUE?	Terminates the program.	Ignores rest of the free-block list and continue execution of <i>fsck</i> . This error condition will always invoke BAD BLKS IN FREE LIST error condition later in Phase 5.

(Continued)

Prompt	n(no)	y(yes)
FIX?	<p> Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p> Replaces count in super-block by actual count.</p>
SALVAGE?	<p> Ignores the error condition. A NO response is only appropriate if the user intends to take other measures to fix the problem.</p>	<p> Replaces actual free-block list with a new free-block list. The new free-block list will be ordered according to the gap and cylinder specs of the -s or -S option to reduce time spent waiting for the disk to rotate into position.</p>

Phase 5 Error Messages

EXCESSIVE BAD BLKS IN FREE LIST (CONTINUE?)

The free-block list contains more than a tolerable number (usually 10) of blocks with a value less than the first data block in the filesystem or greater than the last block in the filesystem.

EXCESSIVE DUP BLKS IN FREE LIST (CONTINUE?)

The free-block list contains more than a tolerable number (usually 10) of blocks claimed by i-nodes or earlier parts of the free-block list.

BAD FREEBLK COUNT

The count of free blocks in a free-list block is greater than 50 or less than 0. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

X BAD BLKS IN FREE LIST

X blocks in the free-block list have a block number lower than the first data block in the filesystem or greater than the last block in the filesystem. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

X DUP BLKS IN FREE LIST

X blocks claimed by i-nodes or earlier parts of the free-list block were found in the free-block list. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

X BLK(S) MISSING

X blocks unused by the filesystem were not found in the free-block list. This error condition will always invoke the BAD FREE LIST condition later in Phase 5.

FREE BLK COUNT WRONG IN SUPERBLOCK (FIX?)

The actual count of free blocks does not match the count in the super-block of the filesystem.

BAD FREE LIST (SALVAGE?)

This message is always preceded by one or more of the Phase 5 information messages. If the *-q* option is specified, the free-block list will be salvaged automatically.

Phase 6: Salvage Free List

This phase reconstructs the free-block list. It has one possible error condition that results from bad blocks-per-cylinder and gap values.

*Phase 6 Error Messages***DEFAULT FREE-BLOCK LIST SPACING ASSUMED**

This is an advisory message indicating the blocks-to-skip (*gap*) is greater than the blocks-per-cylinder, the blocks-to-skip is less than 1, the blocks-per-cylinder is less than 1, or the blocks-per-cylinder is greater than 500. The values of 7 blocks-to-skip and 400 blocks-per-cylinder are used.

Cleanup Phase

Once a filesystem has been checked, a few cleanup functions are performed. The cleanup phase displays advisory messages about the filesystem and status of the filesystem.

*Cleanup Phase Messages***X files Y blocks Z free**

This is an advisory message indicating that the filesystem checked contained *X* files using *Y* blocks leaving *Z* blocks free in the filesystem.

***** BOOT XENIX (NO SYNC!) *****

This is an advisory message indicating that a mounted filesystem or the root filesystem has been modified by *fsck*. If the XENIXsystem is not rebooted immediately without *sync*, the work done by *fsck* may be undone by the in-core copies of tables the UNIX system keeps. If the *-b* option of the *fsck* command was specified and the filesystem is *root*, a reboot is automatically done.

***** FILE SYSTEM WAS MODIFIED *****

This is an advisory message indicating that the current filesystem was modified by *fsck*.

Name

fsdb - File system debugger.

Syntax

/etc/fsdb special [-]

Description

fsdb can be used to patch up a damaged file system after a crash. It has conversions to translate block and i-numbers into their corresponding disk addresses. Also included are mnemonic offsets to access different parts of an i-node. These greatly simplify the process of correcting control block entries or descending the file system tree.

fsdb contains several error-checking routines to verify i-node and block addresses. These can be disabled if necessary by invoking *fsdb* with the optional - argument or by the use of the O symbol. (*fsdb* reads the i-size and f-size entries from the superblock of the file system as the basis for these checks.)

Numbers are considered decimal by default. Octal numbers must be prefixed with a zero. During any assignment operation, numbers are checked for a possible truncation error due to a size mismatch between source and destination.

fsdb reads a block at a time and will therefore work with raw as well as block I/O. A buffer management routine is used to retain commonly used blocks of data in order to reduce the number of read system calls. All assignment operations result in an immediate write-through of the corresponding block.

The symbols recognized by *fsdb* are:

#	absolute address
i	convert from i-number to i-node address
b	convert to block address
d	directory slot offset
+, -	address arithmetic
q	quit
>, <	save, restore an address
=	numerical assignment
=+	incremental assignment
=-	decremental assignment
="	character string assignment
O	error checking flip flop
p	general print facilities

f	file print facility
B	byte mode
W	word mode
D	double word mode
!	escape to shell

The print facilities generate a formatted output in various styles. The current address is normalized to an appropriate boundary before printing begins. It advances with the printing and is left at the address of the last item printed. The output can be terminated at any time by typing the delete character. If a number follows the **p** symbol, that many entries are printed. A check is made to detect block boundary overflows since logically sequential blocks are generally not physically sequential. If a count of zero is used, all entries to the end of the current block are printed. The print options available are:

i	print as i-nodes
d	print as directories
o	print as octal words
e	print as decimal words
c	print as characters
b	print as octal bytes

The **f** symbol is used to print data blocks associated with the current i-node. If followed by a number, that block of the file is printed. (Blocks are numbered from zero.) The desired print option letter follows the block number, if present, or the **f** symbol. This print facility works for small as well as large files. It checks for special devices and that the block pointers used to find the data are not zero.

Dots, tabs, and spaces may be used as function delimiters but are not necessary. A line with just a new-line character will increment the current address by the size of the data type last printed. That is, the address is set to the next byte, word, double word, directory entry or i-node, allowing the user to step through a region of a file system. Information is printed in a format appropriate to the data type. Bytes, words and double words are displayed with the octal address followed by the value in octal and decimal. A **.B** or **.D** is appended to the address for byte and double word values, respectively. Directories are printed as a directory slot offset followed by the decimal i-number and the character representation of the entry name. I-nodes are printed with labeled fields describing each element.

The following mnemonics are used for i-node examination and refer to the current working i-node:

md	mode
ln	link count
uid	user ID number
gid	group ID number

sz	file size
a#	data block numbers (0 - 12)
at	access time
mt	modification time
maj	major device number
min	minor device number

Examples

386i	prints i-number 386 in an i-node format. This now becomes the current working i-node.
ln=4	changes the link count for the working i-node to 4.
ln+=1	increments the link count by 1.
fc	prints, in ASCII, block zero of the file associated with the working i-node.
2i.fdi	prints the first 32 directory entries for the root i-node of this file system.
d5i.fc	changes the current i-node to that associated with the 5th directory entry (numbered from zero) found from the above command. The first logical block of the file is then printed in ASCII.
512B.p0o	prints the superblock of this file system in octal.
2i.a0b.d7=3	changes the i-number for the seventh directory slot in the root directory to 3. This example also shows how several operations can be combined on one command line.
d7.nm="name"	changes the name field in the directory slot to the given string. Quotes are optional when used with nm if the first character is alphabetic.
a2b.p0d	prints the third block of the current i-node as directory entries.

See Also

fsck(ADM), dir(F), filesystem(F).



Name

`fsname-` Prints or changes the name of a file system.

Syntax

```
fsname [-p] [-s name ] /dev/device
```

Description

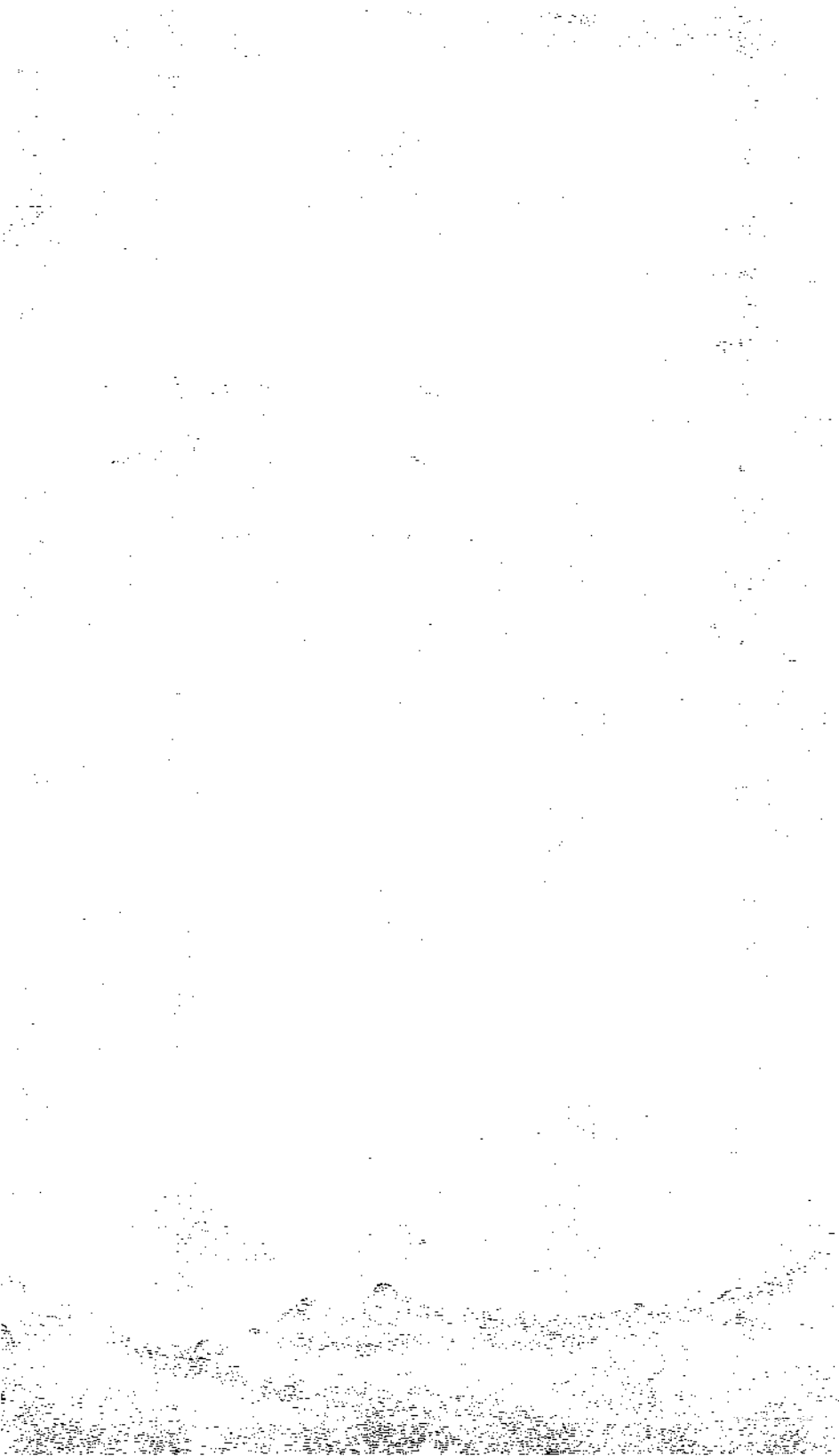
The `/etc/fsname` utility is used to print or change the name of a filesystem. The options are:

- p Select the “pack” name field instead of the filesystem name field.
- s *name* Changes the specified field in the superblock.

The default action is to print the name of the filesystem.

See Also

`mkfs(C)`, `ustat(S)`, `filesystem (F)`



Name

fsphoto - Performs periodic semi-automated system backups

Syntax

fsphoto [-i] schedule [drive]

Description

fsphoto, in conjunction with *fsave* (ADM), provides a semi-automated interface to *backup* (C) for backing-up XENIX filesystems. A human operator is required to mount and dismount tapes or floppies at the appropriate times, so some interaction is necessary, but all such interaction is kept to a minimum to reduce the potential for human error.

The selection and timing of backups for all filesystems is governed by the *schedule* (ADM) database. The system administrator must set up this file, and make arrangements to run *fsphoto* on the implicitly defined schedule (normally once per weekday). *fsphoto* can be invoked most easily from the *sysadmin* (ADM) menu. *fsphoto* interprets *schedule*, and for each filesystem that should be backed-up on that day, runs *fsave* to interact with the operator and backup the filesystem without error.

The optional argument *drive* specifies the magtape or floppy device to use; the default is specified in the *schedule* file.

Backups may be postponed (via *fsave*) or interrupted. The resulting "partial" backups are automatically resumed the next time *fsphoto* is run: Any missed filesystems are backed-up as if the original backup had not been delayed. The -i flag ignores any pending partial backups.

If there is a pending partial backup, the normally scheduled backups are not done. This means that if a partial backup is resumed, and the normally scheduled backups are to be done, *fsphoto* must be run twice.

You must be the super-user to use this program.

Files

/usr/lib/sysadmin/schedule

Database describing which filesystems are to be backed-up when, and at what dump *level*.

/dev/tty

Source of interactive input.

/usr/lib/sysadmin/past

Record of filesystems successfully backed-up in the pending partial backup.

/tmp/backup\$\$

Temporary file for recording successfully backed-up filesystems.

See Also

fsave(ADM), schedule(ADM), backup(C), basename(C)

Diagnostics

fsphoto complains of syntax errors in *schedule*, and exits with a status of 1.

fsphoto complains about illegal or incorrect arguments, and exits with a status of 1.

An interrupt will cause an exit status of 2.

Notes

If a *drive* is explicitly given, the ‘raw’ (/dev/r*) form of the device should be used.

Name

haltsys, reboot - Closes out the file systems and shuts down the system.

Syntax

/etc/haltsys
/etc/reboot

Description

The *haltsys* utility performs a *uadmin()* system call (see *uadmin(S)*) to flush out pending disk I/O, mark the file systems clean, and halt the processor. *haltsys* takes effect immediately, so user processes should be killed beforehand. *shutdown(ADM)* is recommended for normal system shutdown, since it warns users, terminates processes, then calls *haltsys*. Use *haltsys* directly only if you cannot run *shutdown*; for example, because of some system problem.

The *reboot* command performs the same function as *haltsys*, except the system is rebooted automatically afterwards.

Only the super-user can execute *haltsys* or *reboot*.

Notes

haltsys locks hard disk heads.

See Also

shutdn(S), *uadmin(S)*, *shutdown(ADM)*



Name

hdinstall - places newly-created kernel in default location.

Syntax

hdinstall

Description

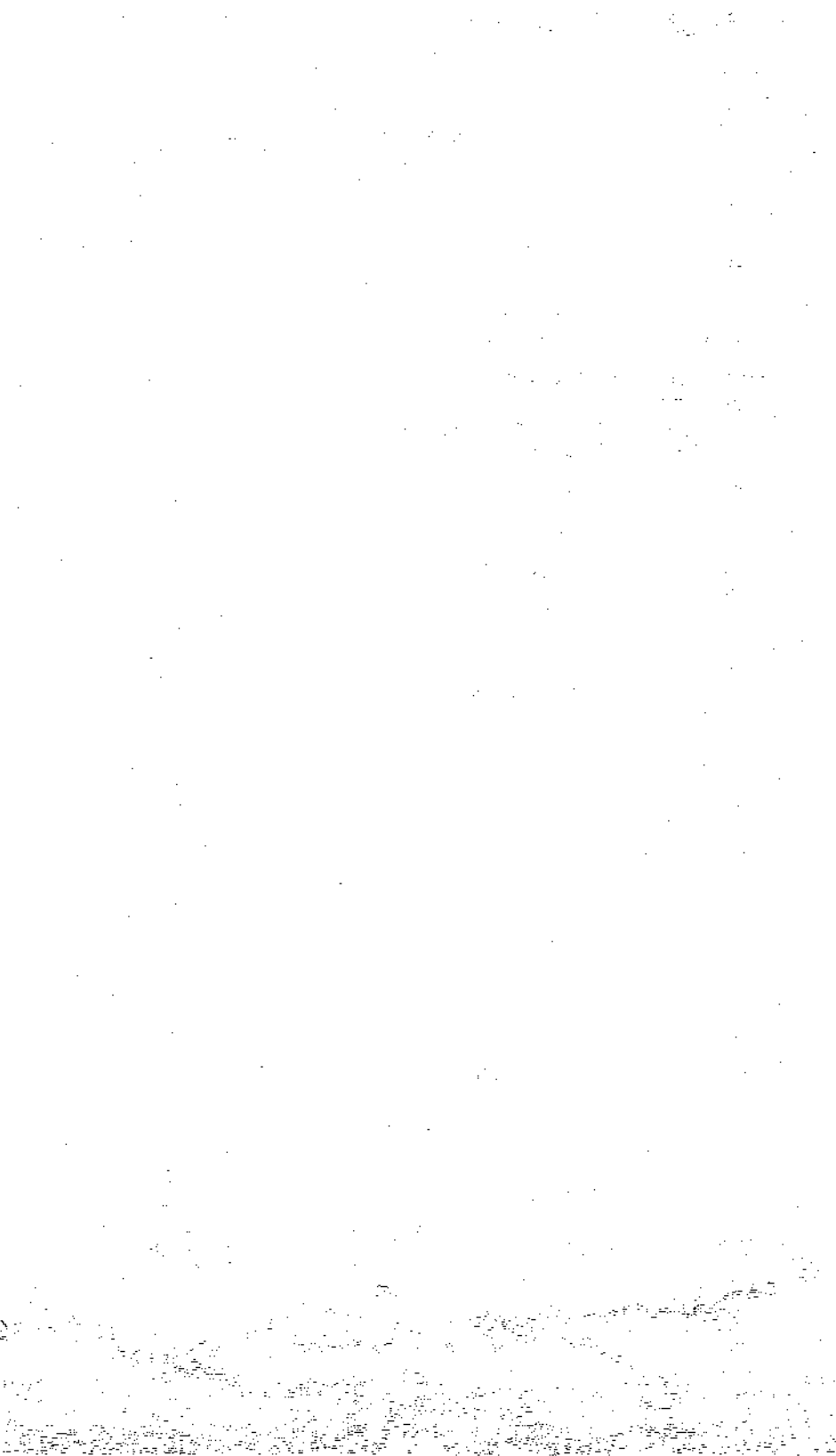
When a new kernel is created with the Link Kit, *hdinstall* must be invoked to place the new kernel in **/xenix**. *hdinstall* moves the “old” */xenix* to a file called */xenix.old* and copies */usr/sys/conf/xenix* to */xenix*, the default location.

Files

/usr/sys/conf/xenix /xenix /xenix.old

See Also

configure(ADM), config(ADM)



Name

idleout - Logs out idle users.

Syntax

idleout [minutes | hours:minutes]

Description

The *idleout* command monitors line activity and logs out users whose terminal remains idle longer than a specified period of time.

The utility uses a default file, */etc/default/idleout*, to indicate the number of hours a user's terminal may remain idle before being logged out. This file has one entry:

```
IDLETIME=time
```

The time format is identical to that used on the command line. The time specified in the default file is overridden by *idletime* if *idletime* is specified on the command line. Note that, if *idletime* is zero, no monitoring takes place and idle users are not logged out.

Files

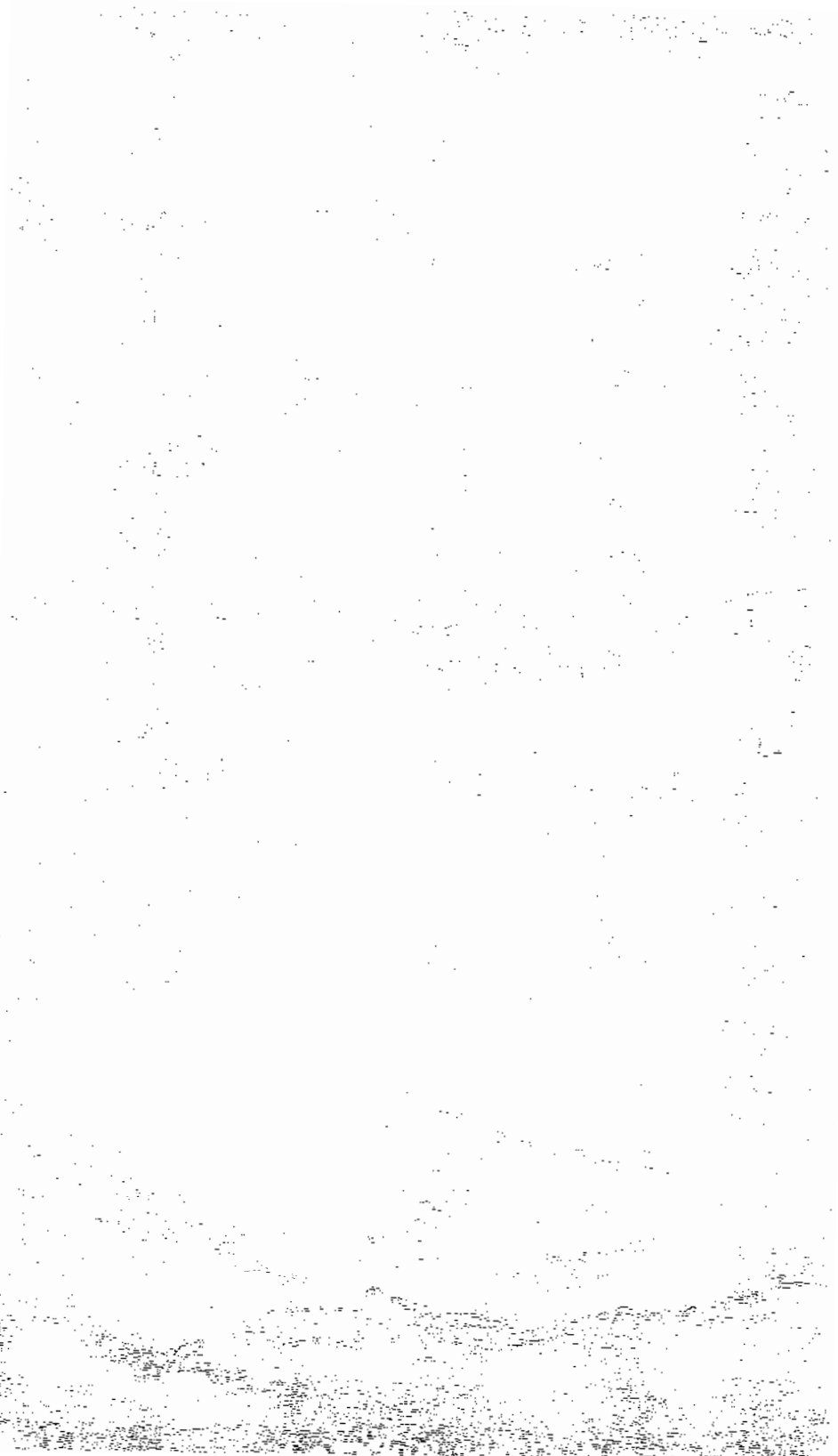
```
/etc/default/idleout  
/etc/utmp  
/etc/wtmp
```

See Also

who(C), *getut(S)*, *kill(S)*

Notes

Only the superuser can run this program.



Name

install - Installation shell script.

Syntax

/etc/install [device]

Description

/etc/install is the *sh*(C) script used to install XENIX distribution (or application program) floppies. It performs the following tasks:

- Prompts for insertion of floppies.
- Extracts files using the *tar*(C) utility.
- Executes **/once/init.*** programs on each floppy after they have been extracted.
- Removes any **/once/init.*** programs when the installation is finished.

The optional argument to the command specifies the device used. The default device is **/dev/install** and is normally linked to **/dev/rfd0**.

Files

/etc/install

/once/init.*



Name

`ipcrm` - Removes a message queue, semaphore set or shared memory ID.

Syntax

`ipcrm` [options]

Description

`ipcrm` removes one or more specified messages, a semaphore or shared memory identifiers. The identifiers are specified by the following *options*:

- q** *msqid* removes the message queue identifier *msqid* from the system and destroys the message queue and data structure associated with it.
- m** *shmid* removes the shared memory identifier *shmid* from the system. The shared memory segment and data structure associated with it are destroyed after the last detach.
- s** *semid* removes the semaphore identifier *semid* from the system and destroys the set of semaphores and data structure associated with it.
- Q** *msgkey* removes the message queue identifier, created with key *msgkey*, from the system and destroys the message queue and data structure associated with it.
- M** *shmkey* removes the shared memory identifier, created with key *shmkey*, from the system. The shared memory segment and data structure associated with it are destroyed after the last detach.
- S** *semkey* removes the semaphore identifier, created with key *semkey*, from the system and destroys the set of semaphores and data structure associated with it.

The details of the removes are described in `msgctl(S)`, `shmctl(S)`, and `semctl(S)`. The identifiers and keys may be found by using `ipcs(ADM)`.

See Also

`ipcs(ADM)`, `msgctl(S)`, `msgget(S)`, `msgop(S)`, `semctl(S)`, `semget(S)`, `semop(S)`, `shmctl(S)`, `shmget(S)`, `shmop(S)`

Note

ipcrm cannot be used to remove semaphores created using *creatsem(S)* or to remove shared memory created using *sdget(S)*.

Name

`ipcs` - Reports the status of inter-process communication facilities.

Syntax

`ipcs [options]`

Description

`ipcs` prints certain information about active inter-process communication facilities. Without *options*, information is printed in short format for message queues, shared memory, and semaphores that are currently active in the system. Otherwise, the information that is displayed is controlled by the following *options*:

- q Print information about active message queues.
- m Print information about active shared memory segments.
- s Print information about active semaphores.

If any of the options **-q**, **-m**, or **-s** are specified, information about only those indicated are displayed. If none of the three options are specified, information about all three are displayed.

- b Print biggest allowable size information (maximum number of bytes in messages on queue for message queues, size of segments for shared memory, and number of semaphores in each set for semaphores). See below, for the meaning of columns in a listing.
- c Print creator's login name and group name. See below.
- o Display information on outstanding usage (number of messages on queue, total number of bytes in messages on queue, and the number of processes attached to shared memory segments).
- p Display process number information. (Process ID of last process to send a message and process ID of last process to receive a message on message queues. It displays the process ID of the creating process and the process ID of the last process to attach or detach on shared memory segments.) See below.
- t Print time information. (Time of the last control operation that changed the access permissions for all facilities. Time of last *msgsnd* and last *msgrcv* on message queues, last *shmat* and last *shmdt* on shared memory, and last *semop*(S) on semaphores.) See below.
- a Use all print *options*. (This is a shorthand notation for **-b**, **-c**, **-o**, **-p**, and **-t**.)
- C *corefile*
Use the file *corefile* in place of */dev/kmem*.
- N *namelist*
The argument will be taken as the name of an alternate *namelist* (*/xenix* is the default).

The column headings and the meaning of the columns in an *ipcs* listing are given below; the letters in parentheses indicate the *options* that cause the corresponding heading to appear; **all** means that the heading always appears. Note that these *options* only determine what information is provided for each facility; they do *not* determine which facilities will be listed.

T	(all)	Type of the facility: q message queue; m shared memory segment; s semaphore.
ID	(all)	The identifier for the facility entry. Note that ID is "X" for facilities created using <i>creatsem(S)</i> or <i>sdget(S)</i> .
KEY	(all)	The key used as an argument to <i>msgget</i> , <i>semget</i> , or <i>shmget</i> to create the facility entry. (Note: The key of a shared memory segment is changed to IPC_PRIVATE from when the segment has been removed until all processes attached to the segment detach it.)
MODE	(all)	The facility access modes and flags: The mode consists of 11 characters that are interpreted as follows: The first two characters are: R if a process is waiting on a <i>msgrcv</i> ; S if a process is waiting on a <i>msgsnd</i> ; D if the associated shared memory segment has been removed. It will disappear when the last process attached to the segment detaches it; C if the associated shared memory segment is to be cleared when the first attach is executed; - if the corresponding special flag is not set.

The next 9 characters are interpreted as three sets of three bits each. The first set refers to the owner's permissions; the next to permissions of others in the user-group of the facility entry; and the last to all others. Within each set, the first character indicates permission to read, the second character indicates permission to write or alter the facility entry, and the last character is currently unused.

The permissions are indicated as follows:

- r** if read permission is granted;
- w** if write permission is granted;
- a** if alter permission is granted;
- if the indicated permission is *not* granted.

OWNER (all)	The login name of the owner of the facility entry.
GROUP (all)	The group name of the group of the owner of the facility entry.
CREATOR (a,c)	The login name of the creator of the facility entry.
CGROUP (a,c)	The group name of the group of the creator of the facility entry.
CBYTES (a,o)	The number of bytes in messages currently outstanding on the associated message queue.
QNUM (a,o)	The number of messages currently outstanding on the associated message queue.
QBYTES (a,b)	The maximum number of bytes allowed in messages outstanding on the associated message queue.
LSPID (a,p)	The process ID of the last process to send a message to the associated queue.
LRPID (a,p)	The process ID of the last process to receive a message from the associated queue.
STIME (a,t)	The time the last message was sent to the associated queue.
RTIME (a,t)	The time the last message was received from the associated queue.
CTIME (a,t)	The time when the associated entry was created or changed.
NATTCH (a,o)	The number of processes attached to the associated shared memory segment.
SEGSZ (a,b)	The size of the associated shared memory segment.
CPID (a,p)	The process ID of the creator of the shared memory entry.
LPID (a,p)	The process ID of the last process to attach or detach the shared memory segment.
ATIME (a,t)	The time the last attach was completed to the associated shared memory segment.
DTIME (a,t)	The time the last detach was completed on the associated shared memory segment.
NSEMS (a,b)	The number of semaphores in the set associated with the semaphore entry.
OTIME (a,t)	The time the last semaphore operation was completed on the set associated with the semaphore entry.

Files

/xenix	system namelist
/dev/kmem	memory
/etc/passwd	user names
/etc/group	group names

See Also

creatsem(S), msgop(S), sdget(S), semop(S), shmop(S)

Notes

Things can change while *ipcs* is running; the picture it gives is only a close approximation.

Name

ips, *isbs*, *ipbs* - IMAGEN protocol handlers.

Syntax

```
/usr/lib/ips [ options ] [ file ]  
/usr/lib/isbs [ options ] [ file ]  
/usr/lib/ipbs [ options ] [ file ]
```

Description

ips, *isbs*, and *ipbs* are the lowest level of IMAGEN printer software. Each handles a different form of communications. They present a very similar view to higher-level software, allowing such software to be relatively independent of communications method.

ips sends files to the IMAGEN printer using the "sequence packet protocol" over RS-232C serial communication lines. This protocol provides for error detection, retransmission, status reporting, detection of unrecoverable errors, and printer usage accounting. The IMAGEN printer must be configured for "sequenced packet (V1)" at the desired baud rate.

isbs supports the "serial byte stream" communications method over RS-232C serial communication lines. This protocol provides for flow control, but no error detection or correction. The IMAGEN printer must be configured for the "serial byte stream" protocol at the desired baud rate, should use an "EOF" character of **0x04**, a "quote" character of **0x02**, an 8-bit-wide data path, XON/XOFF flow control, and should not ignore unprintable characters.

ipbs supports the "parallel byte stream" communications method, which is essentially similar to *isbs* but works on parallel rather than serial ports. The IMAGEN printer must be configured for the parallel interface, should use an "EOF" character of **0x04**, a "quote" character of **0x02**, and should not ignore unprintable characters.

Neither *isbs* nor *ipbs* provides printer-usage accounting. The following information is common to all these programs.

If no *file* name is given, the standard input is read.

The following *options* are recognized:

-D*string*

Imbed *string* into the document control language of the file being sent. If an unrecoverable failure occurs during transmission and the file can be resent, this string will be resent as well.

-astofile

Save the current printer status in file *stofile*.

-idevice

The IMAGEN printer is connected to XENIX special file *device*. The default *device* is the value of **DEVICE** in the file **/etc/default/imagen**.

-llogfile

Save information on the communications needed to send the input in the file *logfile*.

-r The standard input is rewindable. Normally, the standard input is not assumed to be rewindable. Or, if an explicit *file* was named, that file is not rewindable. Normally, an explicitly named file is assumed to be rewindable.

-s Regardless of what other indications may have been given, use the standard output as the printer.

-pdebug

Sets various debugging values.

The following *options* are recognized by all three programs, but only used by *ips* and *isbs*; *ipbs* ignores them:

-o The line characteristics need to be set up if the output is being sent to the standard output. Normally, the characteristics of the standard output are not changed. Or they do not need to be set up if an explicit device was given with the **-i** option. Normally, the characteristics of named devices are changed.

-Bspeed

The baud rate of the serial communications lines is *speed*. It is assumed that the IMAGEN printer is capable of running at the specified *speed*, and that it has been configured to do so. The default *speed* is the value of **SPEED** in the **/etc/default/imagen** file.

The following *options* are recognized by all three programs, but only used by *ips*; both *isbs* and *ipbs* ignore them:

-Aacctfile

Turns on accounting, and places the accounting information in file *acctfile*. This file should be read by *imacct(C)*. There should be a separate *acctfile* for each IMAGEN printer.

-nuser

Names the user or account to be credited with this printing job. Since these programs are normally run by the spooling system, there is no reasonable default username. Therefore, if accounting is enabled by use of the **-A** option, this option should also be

specified. If it is not, all printing is charged to user “???.”

-h*host*

Names the machine to be credited with originating this printing job. If not specified, first *uname*(S) and then */etc/systemid* are used to determine the name of the local system. If the name of the system cannot be determined, “LOCAL” is used.

The following *option* is not supported, and the **-n** option should be used instead:

-u*uid*

The user identification number of the person to be credited with this printing job.

ips, *isbs*, and *ipbs* all read */etc/default/imagen* to obtain various default settings. The values obtained, and the default values, are:

DEVICE=*/dev/imagen*

The name of the XENIX special file connected to the IMAGEN printer. This can be overridden with the **-i** or **-s** options.

SPEED=9600

The baud rate of the IMAGEN printer. This is meaningful only for *ips* and *isbs*, and can be overridden with the **-B** option.

The values of the default settings can be changed to reflect the local system configuration. If */etc/default/imagen* does not exist or cannot be read, the above default values are used.

Files

/dev/imagen

Default name of the XENIX special file connected to the IMAGEN printer.

/dev/null

Default *stsf*file. See the **-a** option.

imagen.log

Default *logfile*. See the **-l** option.

See Also

imagen(M), *ipr*(C)

Author

IMAGEN Corporation.



Name

kbmode - Set keyboard mode or test keyboard support

Syntax

kbmode command [file]

Description

This command can be used to determine if your system keyboard supports AT mode. If it does, this utility can change the keyboard mode in between AT mode and PC/XT compatibility mode.

If the **file** argument is specified, it should be a tty device of one of the multiscreens of the keyboard's group.

Valid commands are:

- test - determine if keyboard supports AT mode
- at - set keyboard to AT mode
- xt - set keyboard to PC/XT compatibility mode

Notes

Some keyboards look like AT keyboard but do not support AT mode. Setting such a keyboard to AT mode will render it useless, unless it can be set to XT mode from another (serial) terminal.

See Also

keyboard(HW)



Name

lpadmin - Configures the lineprinter spooling system.

Syntax

```
/usr/lib/lpadmin -p printer [ options... ]
/usr/lib/lpadmin -x dest
/usr/lib/lpadmin -d[dest]
```

Description

lpadmin configures the lineprinter spooling system to describe printers, classes, and devices. It is used to add and remove destinations, change membership in classes, change devices for printers, change printer interface programs, and to change the system default destination. System managers may also use *lpinit*(ADM) to add new printing destinations to the system. *lpadmin* may not be used when the lineprinter scheduler, *lpsched*(ADM), is running, except where noted below.

Exactly one of the **-p**, **-d**, or **-x** options must be present for every legal invocation of *lpadmin*.

- d[dest]** Makes *dest*, an existing destination, the new system default destination. If *dest* is not supplied, then there is no system default destination. This option may be used when *lpsched*(ADM) is running. No other *options* are allowed with **-d**.
- xdest** Removes destination *dest* from the LP system. If *dest* is a printer and is the only member of a class, then the class will be deleted, too. No other *options* are allowed with **-x**.
- pprinter** Names a *printer* to which all of the *options* below refer. If *printer* does not exist then it will be created.

The following *options* are only useful with **-p** and may appear in any order. For ease of discussion, the printer will be referred to as *p* below.

- cclass** Inserts printer *p* into the specified *class*. *Class* will be created if it does not already exist.
- eprinter** Copies an existing *printer*'s interface program to be the new interface program for *p*.

- h** Indicates that the device associated with *p* is hardwired. This *option* is assumed when creating a new printer unless the **-l** *option* is supplied.
- iinterface** Establishes a new interface program for *p*. *Interface* is the pathname of the new program.
- l** Indicates that the device associated with *p* is a login terminal. The lineprinter scheduler, *lpsched*(ADM), disables all login terminals automatically each time it is started. Before re-enabling *p*, its current *device* should be established using *lpadmin*.
- mmodel** The model printer interface program, *dumb*, is supplied with XENIX lineprinter software. It is a shell procedure which interfaces *lpsched*(ADM) and print devices. It can be found in the directory */usr/spool/lp/model* and may be used as is with *lpadmin -m* or *lpinit*(ADM). This program is an interface for a line printer without special functions and protocol. Form feeds are assumed. System managers may modify copies of *dumb* and then use *lpadmin -i* to associate the copies with printers.
- rclass** Removes printer *p* from the specified *class*. If *p* is the last member of the *class*, then the *class* will be removed.
- vdevice** Associates a new *device* with printer *p*. *Device* is the pathname of a file that is writable by the system manager, *lp*. Note that there is nothing to stop a system manager from associating the same *device* with more than one printer. If only the **-p** and **-v** *options* are supplied, then *lpadmin* may be used while the scheduler is running.

Restrictions

When creating a new printer, the **-v** option and one of the **-e**, **-i**, or **-m** options must be supplied. Only one of the **-e**, **-i**, or **-m** options may be supplied. The **-h** and **-l** keyletters are mutually exclusive. Printer and class names may be no longer than 14 characters and must consist entirely of the characters A - Z , a - z , 0 - 9 and _ (underscore).

Models

Model printer interface programs are shell procedures which interface between *lpsched*(ADM) and devices. Models reside in the directory */usr/spool/lp/model* and may be used as is with *lpadmin -m*. Models should have 644 permission if owned by *lp & bin*, or 664 permission if owned by *bin & bin*. System managers may modify copies of models and then use *lpadmin -i* to associate them with printers. If printers have special options, these can be included in the interface program.

Users can then choose an option with the *lp -o* command.

One model interface program is supplied with XENIX lineprinter software: *dumb*. This is an interface program for a lineprinter without special functions and protocol. Form feeds are assumed. This is a good model for system managers to copy and modify.

Serial printers that need delays or other special *stty*(C) options (such as mapping CR to newline) should have this string included in the model interface program:

```
stty [ options ... ] 0<&1
```

Files

*/usr/spool/lp/**

See Also

accept(C), *enable*(C), *lp*(C), *lpinit*(ADM), *lpsched*(ADM), *lpstat*(C)



Name

lpinit - Adds, reconfigures and maintains lineprinters.

Syntax

`/etc/lpinit`

Description

lpinit is a shell script for configuring and adding new lineprinters to a system, and for maintaining and reconfiguring existing printers. It should only be executed by the system manager.

lpinit asks a series of questions for which the default answers are displayed. You can press RETURN to accept the default value or type in a new value.

lpinit displays a menu with the following options:

- 1) Add a new printer
- 2) Remove a printer
- 3) Reconfigure an existing printer
- 4) Assign a system default printer
- 5) Print lp status information

When reconfiguring an existing printer the following options are given:

- 1) Insert a printer into a class
- 2) Remove a printer from a class
- 3) Install a new interface program for a printer
- 4) Associate a new device with a printer

Information which the system manager may be asked to supply includes:

- The printer device (e.g. `/dev/lp0`).
- The printer character mode. (The default value is *non-interpretive*. See "Notes" below for more information.)
- The printer name (default is *printer*).
- The pathname of the interface program (several example programs are supported).
- The name of a class into which to insert or remove a printer.
- Whether the printer being added or reconfigured is a parallel, serial, or remote printer.
- Whether the printer being added or reconfigured requires special handling for carriage returns and line feeds.

The printer name can be any combination of up to 14 alphanumeric characters or underscores. A printer interface program can be a shell script, C program, or any executable program; or the model interface

program, `/usr/spool/lp/model/dumb`, can be copied and modified. (See the “Models” section of the `lpadmin(ADM)` manual page.)

When adding a new printer, `lpinit` changes the acceptance status of the new lineprinter to “accept,” and enables it to print files. `/etc/lpinit` then asks if the new printer will be the default printing destination. All nonspecific print requests are routed to the default destination (see `lp(C)`).

If the line printer scheduler is running when `lpinit` is invoked, the user is reminded that any jobs which are printing may be interrupted and the user is asked if he wants to continue. The scheduler is restarted when `lpinit` exits only if it was running when `lpinit` was invoked or if a new printer was added.

The steps to configure a new printer can be taken separately, (see `lpadmin(ADM)`, `accept(C)`, `enable(C)`, and `lpsched(ADM)` for more information).

Files

`/etc/lpinit` `/usr/lib/mkdev/lp`

Notes

Some printers require conversions for line-feeds, tabs and form-feeds. In *interpretive* mode, the system sends line-feeds as carriage-returns, tabs as the appropriate number of spaces, and form-feeds as the appropriate number of carriage-returns. In *non-interpretive* mode (the default value), the system sends every character to the printer unmodified.

If you are adding a parallel printer you are asked, after the menu of interface scripts, if the printer requires conversions for line-feed, tab and form-feed. If the printer does not, press RETURN. If the printer does, press `y`. This selects *interpretive* mode and assigns the device `/dev/lp[012]f` to the printer.

If you choose *interpretive* mode, note the following:

You must be sure that the printer's actual top-of-form corresponds to top-of-form as interpreted by the printer driver.

If you run a program that does any non-standard line spacing, such as half-line feeds or 8 lines per inch, the printer's top-of-form will be out of place in subsequent output.

If your output contains characters that are not uniformly spaced, the tab translation may not work properly.

Note that if your printer can be set (for example, with dip switches) to treat line-feed as newline and carriage-return as carriage-return (without a line-feed), and if the printer can do its own tabs and form-feeds, you should select *non-interpretive* mode. If your printer cannot automatically do tabs, you can still use *non-interpretive* mode by using the **-e** option of the *pr(C)* command when printing files that contain tabs.

See Also

accept(C), enable(C), lp(C), lpadmin(ADM), lpsched(ADM), pr(C)



Name

lpsched, *lpshut*, *lpmove* - Starts/stops the lineprinter request scheduler and moves requests.

Syntax

```
/usr/lib/lpsched  
/usr/lib/lpshut  
/usr/lib/lpmove requests destinations  
/usr/lib/lpmove dest1 dest2
```

Description

lpsched schedules requests taken by *lp*(C) for printing on lineprinters.

lpshut shuts down the lineprinter scheduler. All printers that are printing at the time *lpshut* is invoked will stop printing. Requests that were printing at the time a printer was shut down will be reprinted in their entirety after *lpsched* is started again. All lineprinter commands perform their functions even when *lpsched* is not running.

lpmove moves requests that were queued by *lp*(C) between lineprinter destinations. This command may be used only when *lpsched* is not running. The first form of the command moves the named *requests* to the lineprinter *destinations*, *dest*. *Requests* are request IDs as returned by *lp*(C). The second form moves all requests for destination *dest1* to destination *dest2*. As a side effect, *lp*(C) will reject requests for *dest1*.

Note that *lpmove* never checks the acceptance status for the new destination when moving requests (see *accept*(C)).

Files

/usr/spool/lp/*

See Also

accept(C), *enable*(C), *lp*(C), *lpadmin*(ADM), *lpinit*(ADM), *lpstat*(C)



Name

makekey - Generates an encryption key.

Syntax

`/usr/lib/makekey`

Description

makekey improves the usefulness of encryption schemes by increasing the amount of time required to search the key space. It reads 10 bytes from its standard input, and writes 13 bytes on its standard output. The output depends on the input in a way that is intended to be difficult to compute (i.e., to require a substantial fraction of a second).

The first 8 input bytes (the *input key*) can be arbitrary ASCII characters. The last 2 input bytes (the *salt*) are best chosen from the set of digits, dot (.), slash (/), and uppercase and lowercase letters. The *salt* characters are repeated as the first 2 characters of the output. The remaining 11 output characters are chosen from the same set as the *salt* and constitute the *output key*.

The transformation performed is essentially the following: the *salt* is used to select one of 4,096 cryptographic machines all based on the National Bureau of Standards DES algorithm, but broken in 4,096 different ways. Using the *input key* as the key, a constant string is fed into the machine and recirculated. The 64 bits that come out are distributed into the 66 *output key* bits in the result.

See Also

passwd(F)



Name

mkdev - Calls scripts to add peripheral devices.

Syntax

```
/etc/mkdev lp  
/etc/mkdev hd  
/etc/mkdev serial  
/etc/mkdev fs [ device file ]  
/etc/mkdev fd  
/etc/mkdev tape  
/etc/mkdev shl  
/etc/mkdev mouse
```

Description

mkdev calls the scripts to create the requested type of device file(s). *mkdev* may call *lpinit*(ADM), or any other script found in the directory */usr/lib/mkdev*. If no arguments are listed, *mkdev* prints a usage message.

/etc/mkdev lp creates device files for use with line printers. (See *lpinit*(ADM).)

/etc/mkdev hd creates device files for use with a peripheral hard disk. The device files for an internal hard disk already exist. *hdinit* invokes the following utilities: *dparam*(ADM), *badtrk*(ADM), *fdisk*(ADM), and *divvy*(ADM).

/etc/mkdev serial creates device files for use with serial cards. The device files for the first and second ports already exist. Additional device files must be created for the ports added when expansion cards are added to the system. The */etc/ttys* and */etc/ttytype* files are updated.

/etc/mkdev fs performs the system maintenance tasks required to add a new filesystem to the system once the device is created (*mknod*(C)) and the filesystem is made (*mkfs*(ADM)). It creates the */file* and */file/lost+found* directories, reserves slots in the *lost+found* directory, (if either already exist, they are used unmodified) and modifies */etc/checklist*, */etc/default/filesys* and */etc/default* to check (*fsck*(ADM)) and mount (*mount*(ADM), *mmt*(C), *rc*(C)) the filesystem as appropriate. It is usually used in conjunction with *mkdev hd* when adding a second hard disk to the system or with *mkdev fd* when creating a mountable filesystem on a floppy, but can be used on any additional filesystem (for example, on a large internal hard disk).

/etc/mkdev fd creates bootable and root file system floppy disks. The three basic options are: boot and root on a single disk (96 or 135 tpi only), boot and root pair (48 tpi) or filesystem only. Use with *mkdev fs* when creating a filesystem-only floppy.

Several boot and/or root floppies can be created during a single *mkdev fd* session, but *mkdev* does not display a prompt to remove the first floppy and insert the next one. Insert the next floppy when *mkdev* prompts "Would you like to format the floppy first? (y/n)."

/etc/mkdev tape configures the tape driver in preparation for linking a new kernel that includes tape support. It adds a standard quarter-inch cartridge tape driver and/or a mini-cartridge tape driver.

The current driver configurations can be displayed, and changed if necessary. A zero in any of the fields means the driver automatically detects the type of tape device installed and uses the built-in values for that device. If the autoconfiguration values are not correct for your drive, refer to your hardware manual for the correct values, configure the driver and relink the new kernel. *mkdev tape* can also be used to remove a tape driver from the existing kernel.

/etc/mkdev shl initializes necessary devices and configures kernel parameters associated with the number of shell layers sessions available on the system.

/etc/mkdev mouse initializes necessary devices and configures the system to use any supported mouse.

Once the driver is configured, you are prompted for re-linking the kernel. The appropriate devices in */dev* are created.

The various *init* scripts prompt for the information necessary to create the devices.

Files

*/usr/lib/mkdev/**

See Also

badtrk(ADM), *divvy*(ADM), *dparam*(ADM), *fd*(HW), *fdisk*(ADM), *fileys*(F), *format*(C), *hd*(HW), *lp*(HW), *lpinit*(ADM), *mkfs*(ADM), *mknod*(C), *mount*(ADM), *serial*(HW), *usemouse*(C), *tape*(HW).

Notes

Note that */etc/mkdev tape* does not add a tape driver for SCSI tape systems. The appropriate drivers already exist in the SCSI kernel.

Name

mkfs - Constructs a file system.

Syntax

```
/etc/mkfs [-y] [-n] special blocks[: inodes] [gap inblocks]
/etc/mkfs [-y] [-n] special proto [gap inblocks]
           [-s blocks[: inodes]]
```

Description

mkfs constructs a file system by writing on the special file *special*, according to the directions found in the remainder of the command line.

If it appears that the special file contains a file system, operator confirmation is requested before overwriting the data. The *-y* "yes" option overrides this, and writes over any existing data without question. The *-n* option causes *mkfs* to terminate without question if the target contains an existing file system. The check used is to read block one from the target device (block one is the super-block) and see whether the bytes are the same. If they are not, this is taken to be meaningful data and confirmation is requested.

If the second argument is given as a string of digits, *mkfs* builds a file system with a single empty directory on it. The size of the file system is the value of *blocks* interpreted as a decimal number. The boot program is left uninitialized. If the number of inodes is specified, then this number should be the same as the estimated number of files in the file system. If the optional number of inodes is not given, the number of inodes is calculated as a function of the system file size.

If the second argument is a file name that can be opened, *mkfs* assumes it to be a prototype file, *proto*, and takes its directions from that file. The prototype file contains tokens separated by spaces or newlines. The first token is the name of a file to be copied onto block zero as the bootstrap program. The bootstrap program specified should already be stripped of the header (see *strip*(CP)). If the header has not been stripped from the bootstrap program, then *mkfs* issues a warning. The second token is a number specifying the size of the created file system. Typically, it will have been the number of blocks on the device, perhaps diminished by space for swapping. The next token is the *i*-list size in blocks. The next set of tokens comprise the specification for the root file. File specifications consist of tokens giving the mode, the user ID, the group ID, and the initial contents of the file. The syntax of the contents field depends on the mode.

The mode token for a file is a 6 character string. The first character specifies the type of the file. (The characters **-bcd** specify regular, block special, character special and directory files respectively.) The second character of the type is either **u** or **-** to specify set-user-ID mode or not. The third is **g** or **-** for the set-group-ID mode. The rest of the mode is a three digit octal number giving the owner, group, and other read, write, execute permissions; see *chmod(C)*.

Two decimal number tokens come after the mode; they specify the user and group ID's of the owner of the file.

If the file is a regular file, the next token is a pathname whose contents and size are copied. If the file is a block or character special file, two decimal number tokens follow which give the major and minor device numbers. If the file is a directory, *mkfs* makes the entries **.** and **..** and then reads a list of names and (recursively) file specifications for the entries in the directory. The scan is terminated with the token **\$**.

A sample prototype specification follows:

```
/stand/diskboot
4872 110
d--777 3 1
usr d--777 3 1
  sh ---755 3 1 /bin/sh
  ken d--755 6 1
  $
  b0 b--644 3 1 0 0
  c0 c--644 3 1 0 0
  $
$
```

In the second version of the command the **-s** option is a command-line override of the size and number of inodes in the *proto* file.

In both commands, the disk interleaving factors, *gap* and *inblocks* , can be specified. The interleaving factors are a disk hardware function and are described in detail in the *XENIX System Administrator's Guide* .

See Also

chmod(C), *filesystem(F)*, *dir(F)*, *strip(CP)*

Notes

There is no way to specify links when using a prototype file. If the number of inodes is specified on the command line, then the maximum number of inodes in the file system is 65500.

This utility uses BSIZE blocks. Refer to the *machine (HW)* manual page for the size of filesystem blocks.



Name

mkuser - Adds a login ID to the system.

Syntax

`/etc/mkuser`

Description

mkuser is used to add more user login IDs to the system. It is the preferred method for adding new users to the system, since it handles all directory creation and password file updating. To add a new user to the system, *mkuser* requires six pieces of information:

- login name
- user ID
- group ID
- user's login shell
- initial password
- comment string for the `/etc/passwd` file (optional).

The login name is checked against certain criteria (i.e., it must be at least three characters and begin with a lowercase letter). The password must follow standard XENIX conventions (see *passwd(F)*). The password file comment field can be up to 35 characters of information.

mkuser prompts for the shell type to assign to the new user. The selection of shells is determined by the number of shells installed on the system. The shells included in the Run Time System are the standard (Bourne) shell, *sh*, and the restricted shell, *rsh*. Each installed shell is represented by a subdirectory `/usr/lib/mkuser/shell`, which is installed along with the given shell package (see *custom(ADM)*). The shell subdirectory contains the files needed to set up the user's environment to use that shell. These files are `mkuser.defs` and `mkuser.init`, plus any additional files that are specific to a given shell. (For example, `/usr/lib/mkuser/csh/cshrc` and `/usr/lib/mkuser/csh/login` are the standard `.cshrc` and `.login` files used by the *csh* and are copied to the user's home directory when *mkuser* is run.)

mkuser takes some of its parameters from a default file, `/etc/default/mkuser`. An example default file is:

```
HOME=/usr
HOMEMODE=0755
PROFMODE=0640
MAILMODE=0640
```

The HOME entry is the user's home directory, the HOMEMODE entry is the permissions for the user's home directory, the PROFMODE entry is the permissions for the *.login*, *.profile* and *.cshrc* files or other shell-specific files, and the MAILMODE entry specifies the permissions of the user's mailbox.

This file can be edited by the super-user to change these defaults. These defaults can also be defined on a per-shell basis by adding similar entries to the appropriate */usr/lib/mkuser/shell/mkuser.def* file. In addition, there are other files in */usr/lib/mkuser* that can be customized. These include */usr/lib/mkuser/lib/mail*, which is the standard mail message sent to new users, */usr/lib/mkuser/lib/help*, which is the explanation displayed by *mkuser* at startup, */usr/lib/mkuser/shell/mkuser.init*, and any of the shell related files.

mkuser allocates user IDs starting at 200, or the largest number used in the password file. (The operator can also assign a specific user ID to a new user. It must be greater than or equal to 200 and must not already exist.) The default group ID for new users is 50. The minimum group ID allowed for user accounts is 50. The operator is given the choice of assigning the user to the default group or another existing group (only those groups with IDs greater than or equal to 50 are displayed, but any group can be selected). In addition, a new group can be created, in which case the operator may specify the name or ID (or both). If only the name is specified, the next available number is assigned.

mkuser can only be executed by the super-user.

The minimum length of a legal password, and the minimum and maximum number of weeks used in password aging are specified in */etc/default/passwd* by the variables PASSLENGTH, MINWEEKS and MAXWEEKS. For example, these variables might be set as follows:

```
PASSLENGTH=6
MINWEEKS=2
MAXWEEKS=6
```

Files

/etc/passwd

/usr/spool/mail/username

/etc/default/mkuser

/etc/default/passwd

/usr/lib/mkuser/mkuser/lib/help

/usr/lib/mkuser/mkuser/lib/mail

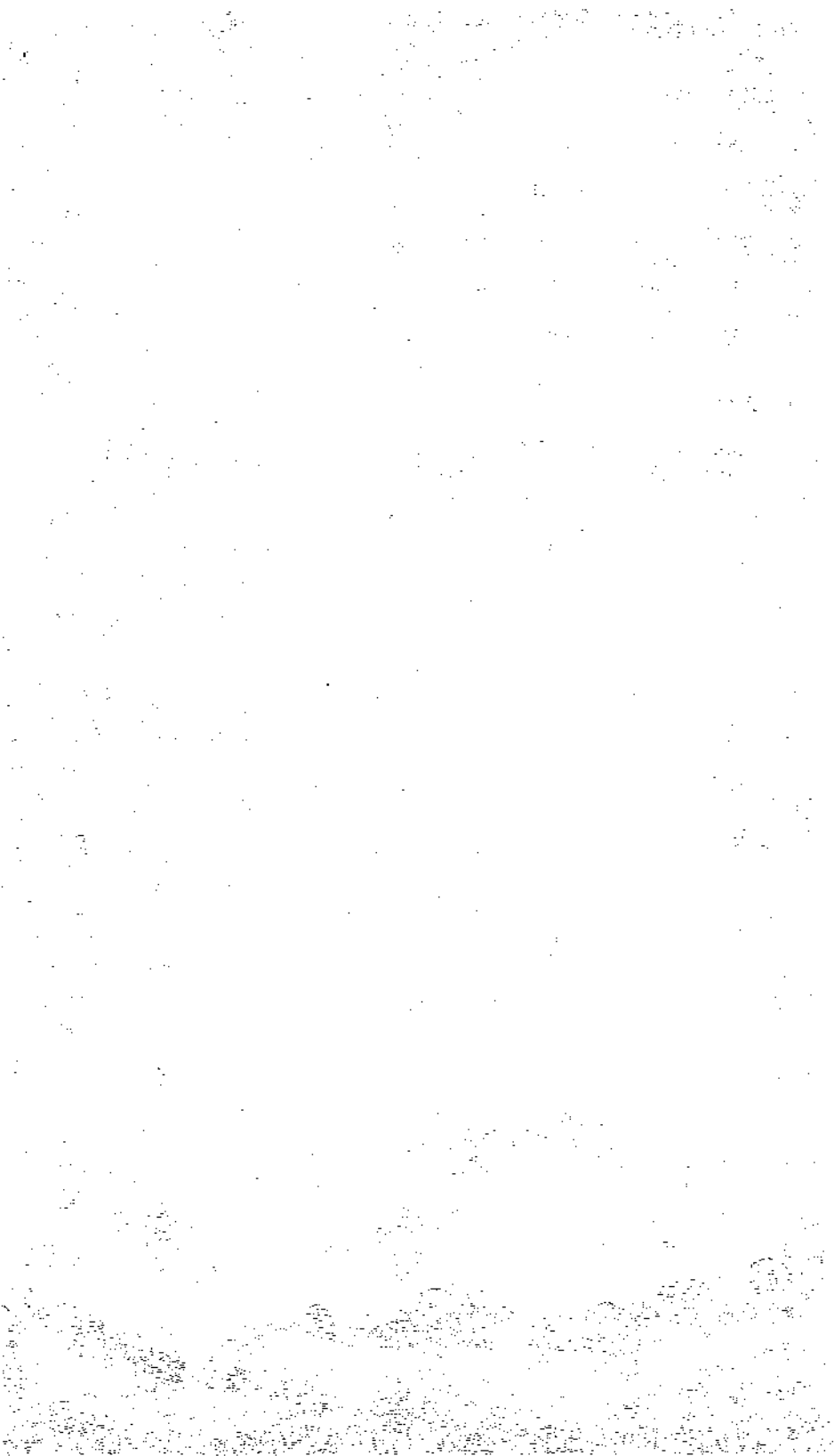
/usr/lib/mkuser/shell/mkuser.defs

/usr/lib/mkuser/shell/mkuser.init

/usr/lib/mkuser/shell/shellfiles

See Also

chmod(C), *custom(ADM)*, *sh(C)*, *csh(C)*, *rsh(C)*, *vsh(C)*, *group(F)*,
passwd(F), *pwadmin(ADM)*, *rmuser(ADM)*



Name

mount - Mounts a file structure.

Syntax

```
/etc/mount [ special-device directory [ -r ] ]
```

Description

mount announces to the system that a removable file structure is present on *special-device*. The file structure is mounted on *directory*. The *directory* must already exist; it becomes the name of the root of the newly mounted file structure. *directory* should be empty. If *directory* contains files, they will appear to have been removed while the *directory* is mounted and reappear when the *directory* is unmounted.

The *mount* and *umount* commands maintain a table of mounted devices. If each special device is invoked without any arguments, *mount* displays the name of the device, and the directory name of the mounted file structure, whether the file structure is read-only, and the date it was mounted.

The optional last argument indicates that the file is to be mounted read-only. Physically write-protected file structures must be mounted in this way or errors occur when access times are updated, whether or not any explicit write is attempted.

umount removes the removable file structure previously mounted on device *special-device*.

Files

/etc/mnttab	Mount table
/etc/default/filesys	Filesystem data

See Also

umount(ADM), mnt(C), mount(S), mnttab(F), default(F)

Diagnostics

mount issues a warning if the file structure to be mounted is currently mounted under another name.

Busy file structures cannot be dismounted with *umount*. A file structure is busy if it contains an open file or some user's working directory.

Notes

Only the super-user can use the *mount* command.

Some degree of validation is done on the file structure, however it is generally unwise to mount corrupt file structures.

Be warned that when in single-user mode, the commands that look in */etc/mnttab* for default arguments (for example *df*, *ncheck*, *quot*, *mount*, and *umount*) give either incorrect results (due to a corrupt */etc/mnttab* from a non-shutdown stoppage) or no results (due to an empty *mnttab* from a *shutdown* stoppage).

When multi-user, this is not a problem; */etc/rc* initializes */etc/mnttab* to contain only */dev/root* and subsequent mounts update it appropriately.

The *mount*(ADM) and *umount*(ADM) commands use a lock file to guarantee exclusive access to */etc/mnttab*. The commands which just read it (those mentioned above) do not, so it is possible that they may hit a window, which is corrupt. This is not a problem in practice since *mount* and *umount* are not frequent operations.

When mounting a file system on a floppy disk you need not use the same *directory* each time. However, if you do, the full pathnames for the files are consistent with each use.

Floppy disks must be unprotected (no write-protect tab) to be mounted as a filesystem. Always **unmount** filesystems on floppy disks before removing them from the floppy drive. Failure to do so requires running **fsck** the next time the disk is **mounted** .

Name

`mmdir` - Moves a directory.

Syntax

`/etc/mmdir` *dirname* *name*

Description

mmdir moves directories within a file system. The directory (*dirname*) must be a directory. If there is already a directory or file with the same name as *name*, **mmdir** fails.

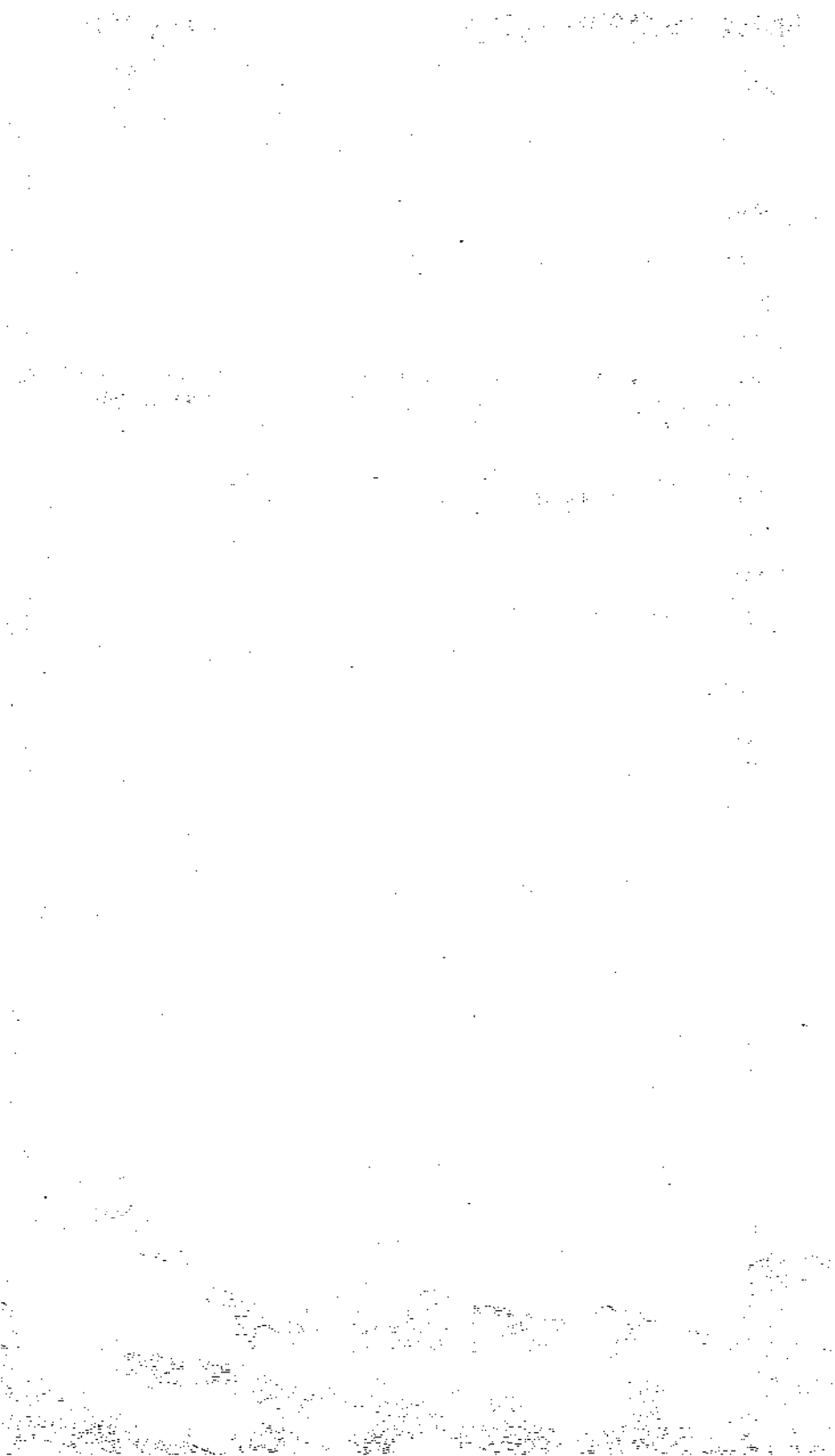
Neither name may be a sub-set of the other. For example, you cannot move a directory named `/x/y` to `/x/y/z`, and vice versa.

Notes

You must be *root* to use *mmdir*.

See Also

`mkdir`(C)



Name

ncheck - Generates names from inode numbers.

Syntax

```
ncheck [ -i numbers ] [ -a ] [ -s ] [ filesystem ]
```

Description

ncheck with no argument generates a pathname and inode number list of all files on the set of file systems specified in */etc/mnttab*. The two characters “/.” are appended to the names of directory files. The **-i** option reduces the report to only those files whose inode numbers follow. The **-a** option allows printing of the names . and .., which are ordinarily suppressed. The **-s** option reduces the report to special files and files with set-user-ID mode; it is intended to discover concealed violations of security policy. A single *filesystem* may be specified rather than the default list of mounted file systems.

Files

/etc/mnttab

See Also

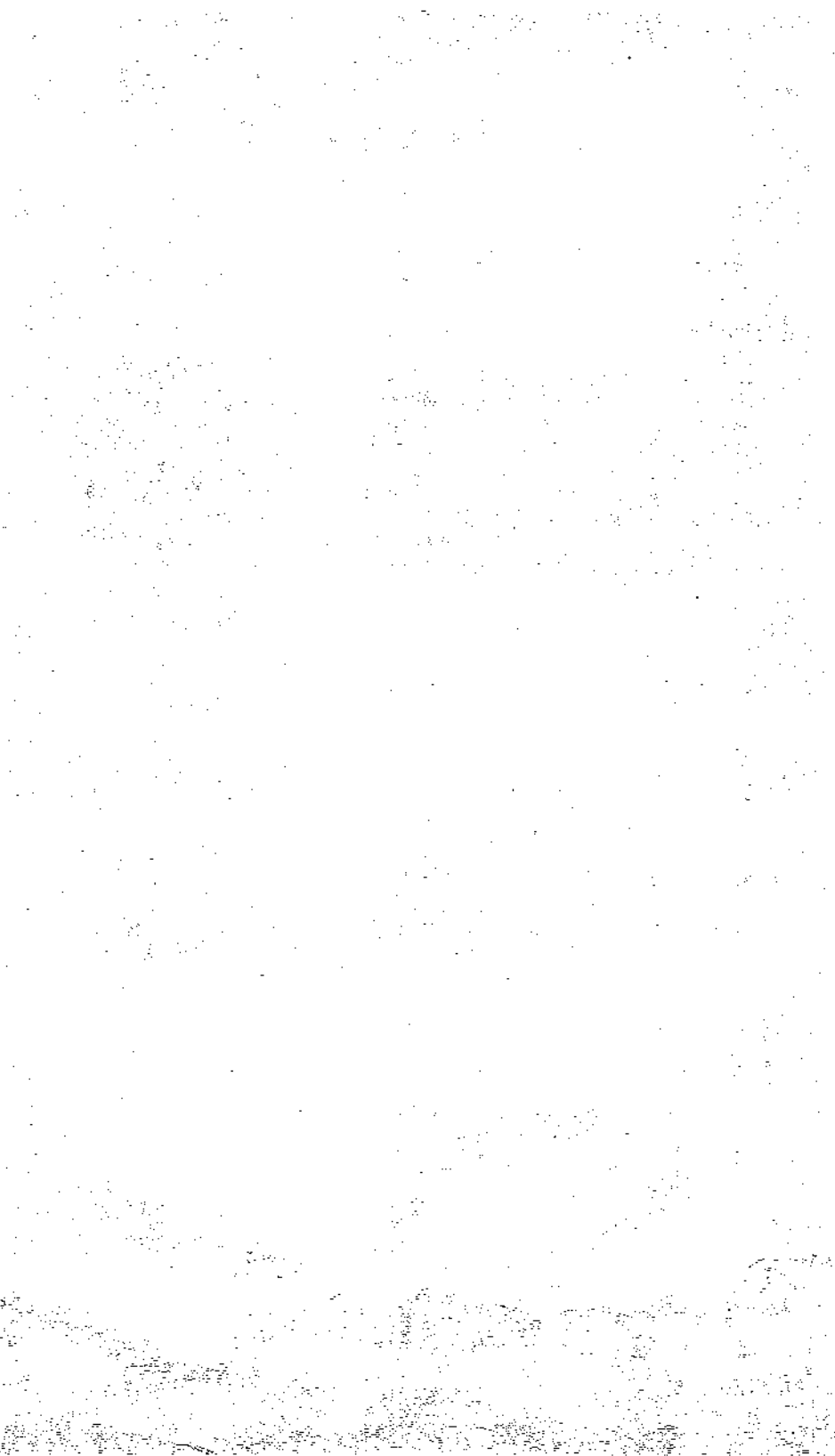
fsck(ADM), sort(C)

Diagnostics

When the file system structure is improper, ?? denotes the ‘parent’ of a parentless file and a pathname beginning with ... denotes a loop.

Notes

See *Notes* under *mount*(ADM).



Name

netutil - Administers the XENIX network.

Syntax

netutil [option] [-x] [-e]

Description

The *netutil* command allows the user to create and maintain a network of XENIX machines. A Micnet network is a link through serial lines of two or more XENIX systems. It is used to send mail between systems with the *mail(C)* command, transfer files between systems with the *rcp(C)* command, and execute commands from a remote system with the *remote(C)* command.

The **netutil** command is used to create and distribute the data files needed to implement the network. It is also used to start and stop the network. The *option* argument may be any one of **install**, **save**, **restore**, **start**, **stop**, or the numbers 1 through 5 respectively. The **-x** option logs transmissions and the **-e** options logs errors. The **-x** and **-e** options work only when they are used in conjunction with **start**, **stop** or their decimal equivalents (4 and 5).

The **install** option interactively creates the data files needed to run the network. The **save** option saves these files on floppy or hard disks, allowing them to be distributed to the other systems in the network. If you save the micnet files to the hard disk, you can then use *uucp(C)* to transfer the files to the other machines. This option specifies the name of the backup device and prompts for whether this is the desired device to use. The user can specify an alternate device, including a file on the hard disk. The name of the default backup device is located in the file `/etc/default/micnet`. This can be changed depending on system configuration. The **restore** option copies the data files from floppy disk back to a system. The **start** option starts the network. The **stop** option stops the network. An *option* may also be any decimal digit in the range 1 to 5. If invoked without an *option*, the command displays a menu from which to choose one. Once an option is selected, it prompts for additional information if needed.

A network must be installed before it can be started. Installation consists of creating appropriate configuration files with the **install** option. This option requires the name of each machine in the network, the serial lines to be used to connect the machines, the speed of transmission for each line, and the names of the users on each machine. Once created, the files must be distributed to each computer in the network with the **save** and **restore** options. The network is started by using the **start** option on each machine in the network. Once started, mail and remote commands can be passed along the network. A record of the transmissions between computers in a network can be kept in the network log files. Installation of the network is described in the *XENIX System Administrator's Guide*.

Files

/bin/netutil
/etc/default/micnet

See Also

aliases(M), aliahash(ADM), mail(C), micnet(F), remote(C), rcp(C), systemid(F), top(F).

Name

`pwadmin` - Performs password aging administration.

Syntax

`pwadmin [-min weeks -max weeks] options`

Description

`pwadmin` is used to examine and modify the password aging information in the password file. The options one can specify are as follows:

- d user** Displays the password aging information.
- f user** Forces the user to change his password at the next login.
- c user** Prevents the user from changing his password.
- a user** Enables password aging for the given user. This option sets the minimum number of weeks that the user must wait before changing his password and the maximum number of weeks that a user can keep his current password for the values defined by the `MINWEEKS` and `MAXWEEKS` variables in the `/etc/default/passwd` file. If the file is not found or the defined values are not in the range 0 to 63, the default values 2 and 4 are used.
- n user** Disables the password aging feature.
- min weeks**
Enables password aging and sets the minimum number of weeks before the user can change his password to *weeks*. (This prevents him from changing his password back to the old one).
- max weeks**
Enables password aging and sets the number of weeks so the user can keep his current password set for *weeks*.

Files

`/etc/passwd`

`/etc/default/passwd`

See Also

`passwd(C)`, `passwd(F)`

Notes

The user must not attempt to force a new password by setting both the **-min** and **-max** values to zero. To force a password, use the **-f** option.

The user must not attempt to prevent further password changes by setting the **-min** value greater than the **-max** value. To prevent changes, use the **-c** option.

Name

`rmuser` - Removes a user account from the system.

Syntax

`/etc/rmuser`

Description

rmuser removes users from the system. It begins by prompting for a user name; after receiving a valid user name as a response, it then deletes the named user's entry in the password file, and removes the user's mailbox file, the **.profile** file, and the entire home directory. It will also remove the users group entry in **/etc/group** if the user was the only remaining member of that group, and the group ID was greater than 50.

Before removing a user ID from the system, make sure its mailbox is empty and that all files belonging to that user ID have been saved or deleted as required.

The *rmuser* program will refuse to remove a user ID or any of its files if one or more of the following checks fails:

- The user name given is one of the "system" user names such as root, sys, sysinfo, cron, or uucp. All user IDs less than 200 are considered reserved for system use, and cannot be removed using *rmuser*. Likewise, all group IDs less than 50 are not removable using *rmuser*.
- The user's mailbox exists and is not empty.
- The user's home directory contains files other than **.profile**.

rmuser can only be executed by the super-user.

Files

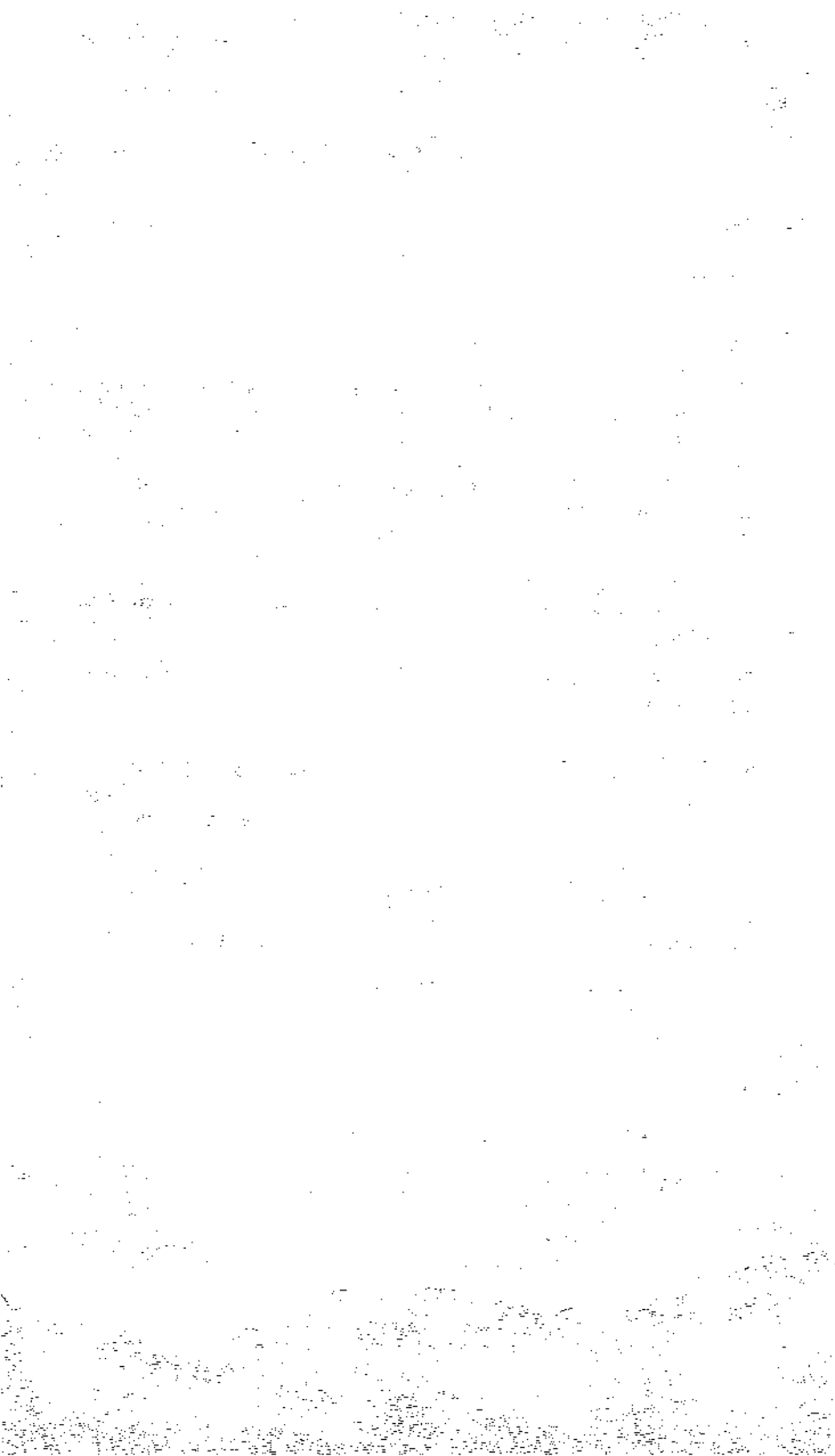
`/etc/passwd`

`/usr/spool/mail/username`

`$HOME`

See Also

`mkuser(ADM)`, `backup(C)`



Name

runbig - Runs a command that may require more memory than normal.

Syntax

runbig command [arguments]

Description

runbig executes commands that may require more memory than is normally available to a user process. While *runbig* is executing the specified *command*, it ignores the restriction on the default of memory available to the user process. The *command* will run normally until it grows to be larger than the amount of memory available to a user process. It is then locked in core memory and not swapped until it either exits or shrinks to a size less than or equal to the size of a default user process.

The removal of the process size restriction during execution of *runbig* will be preserved during an *exec(S)* system call, but not for a *fork(S)* system call.

See Also

exec(S), fork(S)

Notes

Running programs greater than the default process size, and therefore, possibly greater than the size of the disk swap area, may severely impact system performance.

runbig has no effect on systems whose memory size is much less than the size of the disk swap area.



Name

schedule - Database for automated system backups

Description

The *schedule* database is used in conjunction with *fsphoto*(ADM) to partially automate system-wide backups. For each filesystem to be backed-up, a cyclical schedule of *backup*(C) levels is specified.

This cyclical schedule (or *cycle*) is a list of dump levels to perform (including no dump at all) and a pointer to the last-used element of that list. The pointer is advanced to the next element of the list on a regular basis (each time *fsphoto* is run, usually once per day), starting over at the beginning each time it falls off the end. It is advanced, however, only on success - the desired dump must have been successful.

Each entry in the file is on a separate line. Blank and comment lines (beginning with “#”) may be placed anywhere. Several keywords are recognized:

site *sitename*

Sitename is passed to *fsave* as a description to place on each tape label. Usually, *sitename* is the name of the company or a building number.

media *drive k sizes...* [*format*]

Device *drive* is a floppy capable of handling volumes with any of the listed *sizes* (in kilobytes). If specified, *format* is the XENIX command used to format the described floppies.

media *drive d density sizes...* [*format*]

Device *drive* is a *density* BPI magtape capable of handling tapes of any of the indicated *sizes* (in feet). Like floppies, *format* is the optional XENIX command used to format the described tape.

[0-9] *size savetime importance marker*

Description of each dump level, as described in *fsave*(ADM). The defaults are:

Level	Size	Savetime	Importance	Marker
0	-	"1 year"	critical	none
1	-	"3 months"	necessary	none
2...7	-	"1 month"	important	none
8	-	"2 weeks"	useful	none
9	-	"1 week"	precautionary	none

All four fields must be specified. A *size* of - means to use the first size listed in the appropriate **media sizes** list.

Keywords should be placed before any filesystem dump schedules. A filesystem dump schedule is of the form:

/dev/rfilesystem cycle

The filesystem resident on device */dev/filesys* is to be backed-up according to *cycle*, which is a space-separated list of dump levels (the digits **0** to **9**, passed to *dump*), or the letter **x**, meaning no dump should occur.

A dump *cycle* must have at least one member, but it may be of any length. Different filesystems may have *cycles* of different lengths.

Here is a sample *schedule* file:

```
# SYSTEM BACKUP SCHEDULE
site mymachine

# Media Entries
# 96 tpi 1.2 MB floppy 0
# media /dev/rfd096ds15 k 1200 format /dev/rfd096ds15
# 96 tpi 1.2 MB floppy 1
# media /dev/rfd196ds15 k 1200 format /dev/rfd196ds15
# Cartridge tape 0
# media /dev/rct0 d 20000 300 450 600 tape erase
# 9-track tape drive
# media /dev/rmt0 d 1600 2400 1200 600
# Backup Descriptor Table
# Backup Vol. Save for Vitality Label
# level size how long (importance) marker
# 0 - "1 year" critical "a red sticker"
# 1 - "4 months" necessary "a yellow sticker"
# 8 - "3 weeks" useful "a blue sticker"
# 9 - "1 week" precautionary none
# Schedule Table
# 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
# Filesystem MTWTF MTWTF MTWTF MTWTF
/dev/root 0 x 9 x 9 8 x 9 x 9 1 x 9 x 9 8 x 9 x 9
/dev/ru 9 0 9 9 9 9 8 9 9 9 9 1 9 9 9 9 8 9 9 9
```

In the example above, filesystem */dev/root* is dumped using a level 0 dump the first time *fsphoto* is run (on a Monday), and if that dump is successful, the next (second) time it runs (Tuesday), no dump is performed. If doing nothing is successful, the third time run (Wednesday) a level 9 dump occurs. If that dump succeeds, no dump occurs the fourth time (Thursday), but the fifth time *fsphoto* is run (Friday), a level 9 dump is made.

Each time a successful dump at the specified level happens, the pointer advances so that the next run of *fsphoto* (on the next weekday) will do the next dump scheduled for that filesystem. If however, a dump fails (or is interrupted or postponed by the operator) the pointer

is not advanced; hence, the next time *fsphoto* is attempted, the same level dump will again be tried so the sequence will not be broken (but the timing may be off).

Continuing the example, the nineteenth time *fsphoto* runs, a level 9 dump of */dev/rroot* is done, no dump is performed the next (twentieth) time, but the twenty-first time (Monday of every fifth week) the cycle starts over again at the beginning with a level 0 dump.

The larger and more rapidly changing filesystems */dev/ru* is dumped more frequently (each time *fsphoto* is run - once a day - instead of every other time), and the levels used are staggered to prevent having to perform two full-scale dumps (like levels 0 or 1) of the large filesystems on the same day. The backup cycle period is also shorter, two weeks instead of four.

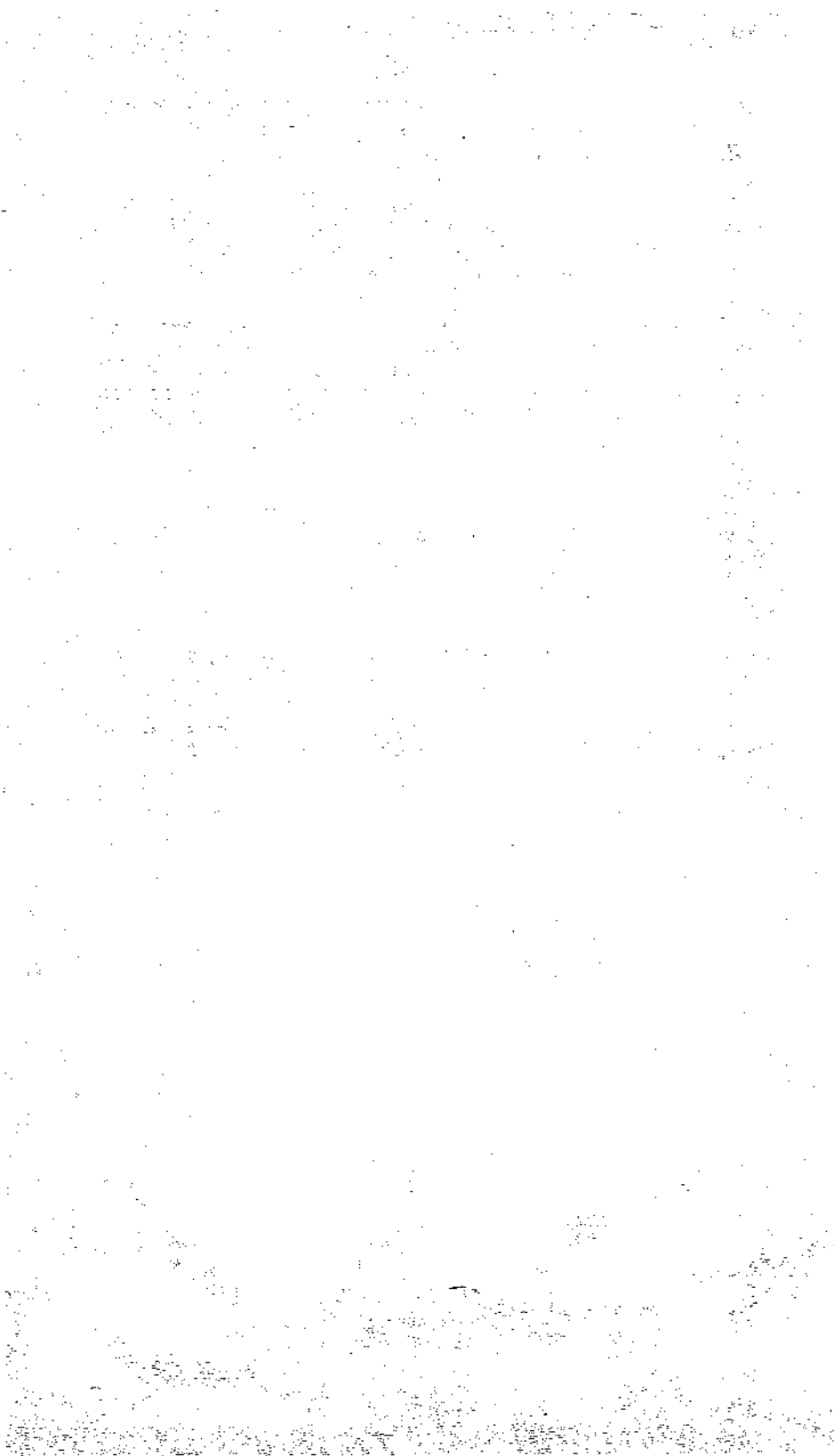
See Also

fsphoto(ADM), *fsave*(ADM), *backup*(C)

Notes

Keywords and filesystem names must not be preceded by any spaces or tabs.

It is not necessary to specify the name of the "raw" (*/dev/r**) device for each filesystem, but the backups are faster if this is done.



Name

setclock - Sets the system real-time (time of day) clock.

Syntax

setclock [*time*]

Description

The **setclock** file sets the battery-powered, real-time time of day clock to the given *time*. If *time* is not given, the current contents of the battery-powered clock are displayed. The *time* must be a combination of digits with the form:

MMddhhmmyy

where *MM* is the month, *dd* is the day, *hh* is the hour, *mm* is the minute, and *yy* is the last two digits of the year. If *yy* is not given, it is taken from the current system time. For example, the command:

082615035

sets the time of day clock to 15:03 on August 26, 1985.

Files

/etc/setclock

See Also

clock(F)

Notes

Not all computers have battery-powered real-time time of day clocks. Refer to your computer's hardware reference manual.



Name

setmnt - Establishes /etc/mnttab table.

Syntax

/etc/setmnt

Description

setmnt creates the **/etc/mnttab** table (see *mnttab*(F)), which is needed for both the *mount*(ADM) and *umount*(ADM) commands. *setmnt* reads the standard input and creates a *mnttab* entry for each line. Input lines have the format:

filesys node

where *filesys* is the name of the file system's *special file* (e.g., "hd0") and *node* is the root name of that file system. Thus *filesys* and *node* become the first two strings in the *mnttab*(F) entry.

Files

/etc/mnttab

See Also

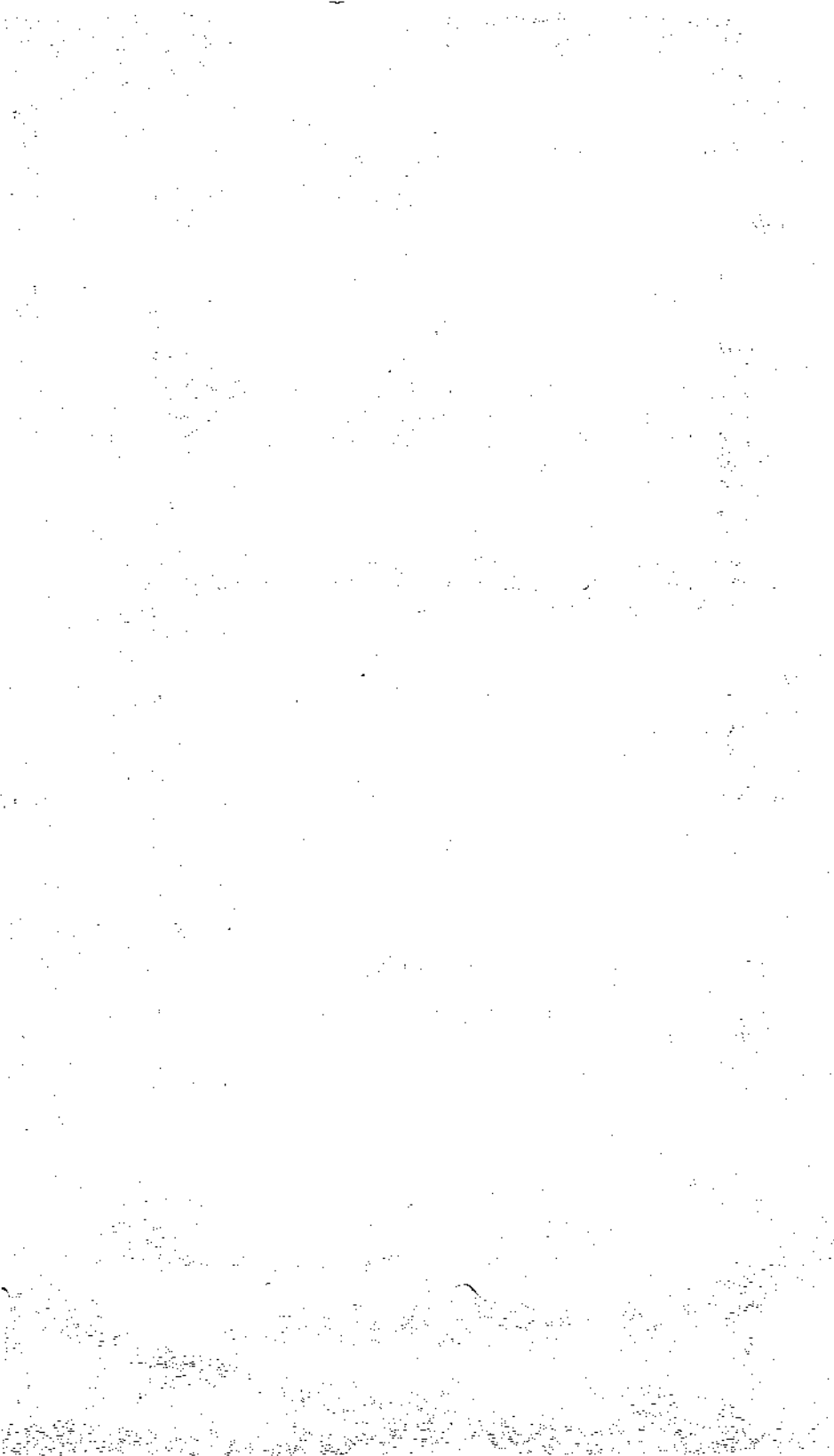
mnttab(F)

Notes

If *filesys* or *node* are longer than 128 characters, errors can occur.

setmnt silently enforces an upper limit on the maximum number of *mnttab* entries.

setmnt is normally invoked by **/etc/rc** when the system boots up.



Name

settime - Changes the access and modification dates of files.

Syntax

```
settime mmddhhmm [ yy ] [ -f fname ] name ...
```

Description

Sets the access and modification dates for one or more files. The dates are set to the specified date, or to the access and modification dates of the file specified via **-f**. Exactly one of these methods must be used to specify the new date(s). The first *mm* is the month number; *dd* is the day number in the month; *hh* is the hour number (24 hour system); the second *mm* is the minute number; *yy* is the last two digits of the year and is optional. For example:

```
settime 1008004583 ralph pete
```

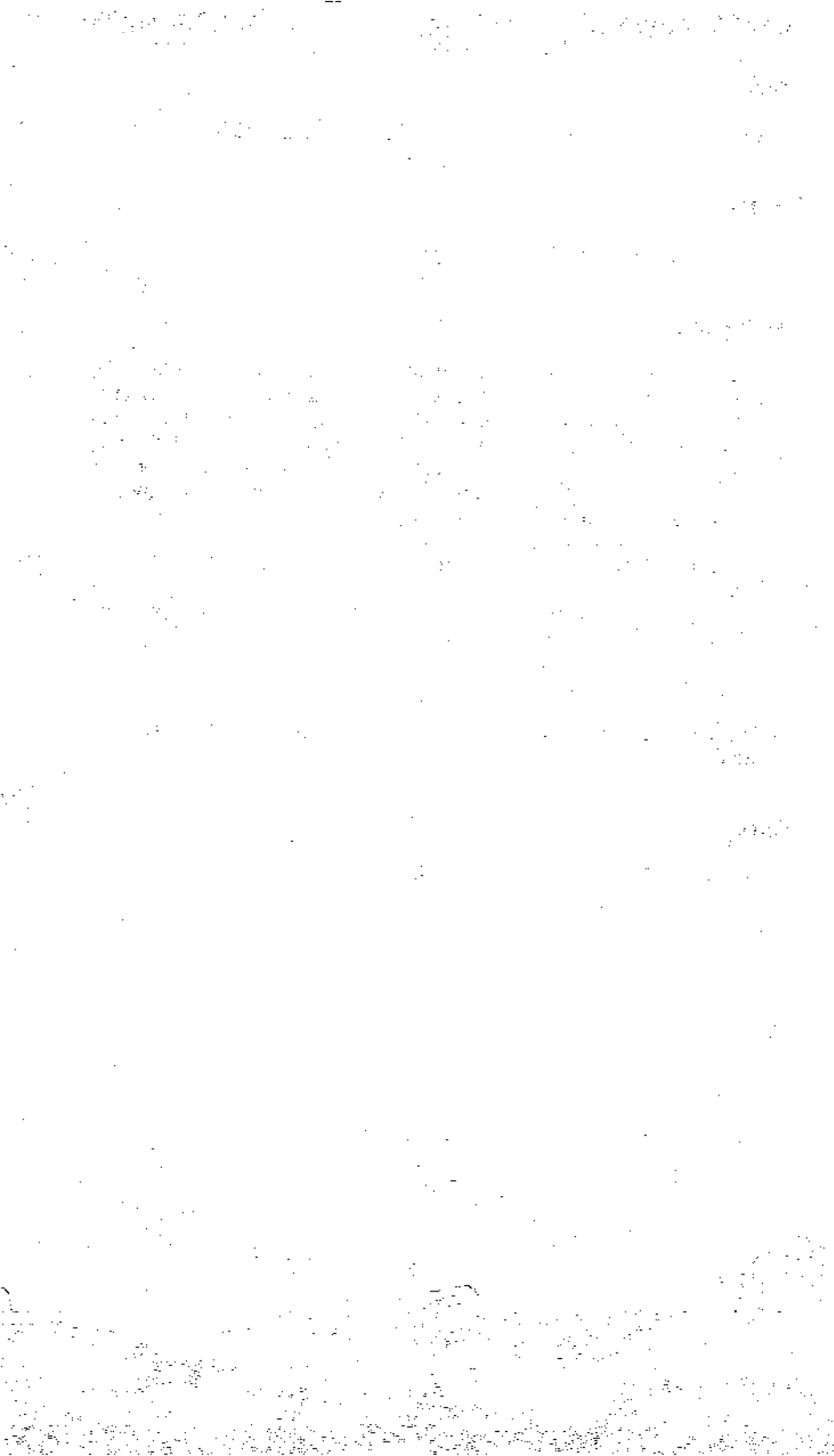
sets the access and modification dates of files *ralph* and *pete* to Oct 8, 12:45 AM, 1983. Another example:

```
settime -f ralph john
```

This sets the access and modification dates of the file *john* to those of the file *ralph*.

Notes

Use of *touch* in place of *settime* is encouraged.



Name

shutdown - Terminates all processing.

Syntax

```
/etc/shutdown [ time ] [ su ]
```

Description

shutdown is part of the XENIX operation procedures. Its primary function is to terminate all currently running processes in an orderly and cautious manner. The *time* argument is the number of minutes before a shutdown will occur. The optional *su* argument lets the user go single-user, without completely shutting down the system. However, the system is shut down for multi-user use. *shutdown* goes through the following steps. First, all users logged on the system are notified to log off the system by a broadcasted message. All file system superblocks are updated before the system is stopped (see *sync*(ADM)). This must be done before rebooting the system, to ensure file system integrity.

You must be super-user to execute the *shutdown* command.

See Also

sync(ADM), *umount*(ADM), *wall*(ADM), *boot*(HW)

Diagnostics

The most common error diagnostic that will occur is *device busy*. This diagnostic appears when a particular file system could not be unmounted. See *umount*(ADM).

Notes

Once *shutdown* has been invoked, it must be allowed to run to completion and must *not* be interrupted by pressing BREAK or DEL.

shutdown does not work when executed from within a shell layer.

shutdown locks the hard disk heads.



Name

sync - Updates the super-block.

Syntax

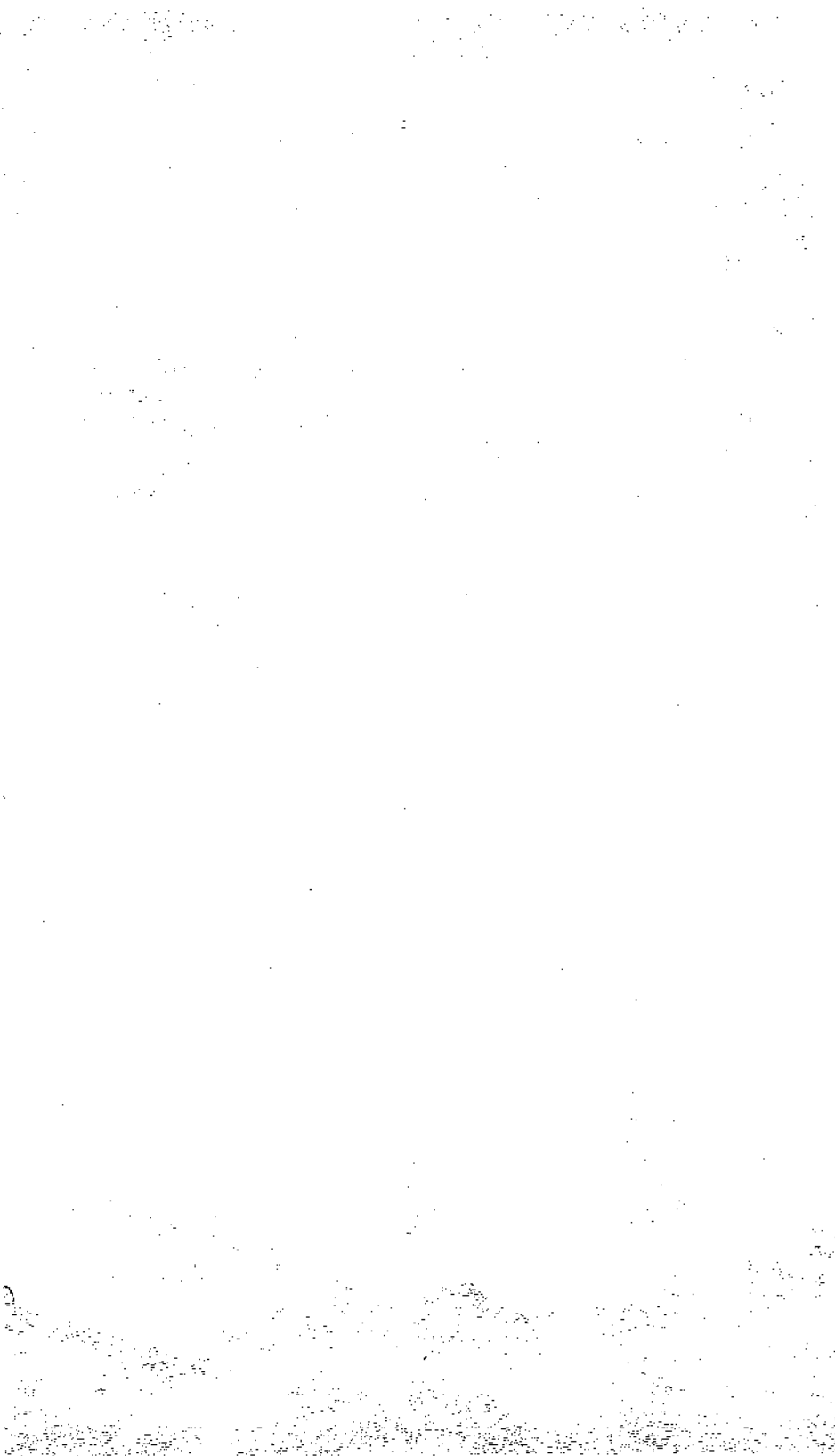
sync

Description

sync executes the *sync* system primitive. If the system is to be stopped, *sync* must be called to ensure file system integrity. Note that *shutdown*(ADM) automatically calls *sync* before shutting down the system.

See Also

sync(S)



Name

sysadmin - Performs file system backups and restores files.

Syntax

/etc/sysadmin

Description

sysadmin is a utility for performing filesystem backups and for restoring files from backup volumes, and includes several options. Its main function is to act as a front-end for the *fsphoto(ADM)* utility, which performs backups according to an established schedule. Depending on the day of the week, a daily incremental backup (level 9), or a periodic full backup (level 0) is automatically selected. *sysadmin* can also be invoked to do an unscheduled backup. It can provide a listing of the files backed up and also has facilities for restoring individual files and complete filesystems from backups.

The main *sysadmin* menu appears as follows:

Filesystem Maintenance Options

1. Perform a scheduled backup
2. Perform an unscheduled backup
3. List the contents of an archive
4. Restore backed up file(s)
5. Restore an entire filesystem
6. Check backup archive integrity

Enter an option or enter q to quit:

Any supported archive medium may be used to create backups. Any filesystem may be backed up. Menus of these devices are created for each option from the files */tmp/backup.list*, */etc/default/archive*, and */etc/default/filesys*.

You must be the super-user to use this program.

Files

/tmp/backup.list
/etc/default/archive
/etc/default/filesys

See Also

fsphoto(ADM), mkfs(ADM), backup(C), dumpdir(C), restore(C), archive(F), fileys(F)

Notes

/tmp/backup.list, */etc/default/archive* and */etc/default/filesys* may be edited to add devices, or to delete entries for devices that are no longer used.

Warning

You should never backup more than one filesystem to the tape devices */dev/nrct0* and */dev/nrct2*. This is because, although *backup* can write more than one filesystem to */dev/nrct0* or */dev/nrct2*, *restore* may not be able to restore more than one filesystem from these devices.

You must also be sure to close the floppy door when inserting floppy disks during a backup. If you fail to do so in a multi-floppy backup, the entire backup will fail and you will have to begin again.

Name

sysadmsh - Menu driven system administration utility

Syntax

sysadmsh

Description

sysadmsh is an easy-to-use menu interface designed to provide novice users with the tools needed for day-to-day system administration of the XENIX system.

WARNING: *sysadmsh* does not replace the XENIX documentation. It provides an overview of available system administration features and a reminder of tasks which need to be performed regularly. An understanding of the *XENIX Installation Guide*, the *XENIX System Administrator's Guide*, and the *XENIX User's Guide* is necessary to use *sysadmsh*.

Usage

To use this utility enter:

sysadm

at the login prompt. This sets your login shell to be the *sysadmsh* menu hierarchy. You may access many useful commands and sub-menus, all presented in simple, descriptive terms.

Alternately, *sysadmsh* menus may also be invoked by logging in as the super-user (root) and entering:

sysadmsh

at the shell prompt.

Once you are in *sysadmsh*, on-line instructions for its use may be obtained by selecting the <F1> key.

Some *sysadmsh* options must be run from the system console device. Some options must be run while in single user (system maintenance) mode. Check the documentation manual page referenced by the menu selection for more information.

Files

See Also

XENIX System Administrator's Guide
XENIX User's Guide
XENIX Installation Guide

acctcom(ADM), accton(ADM), alias(M), asktime(ADM), at(C), badtrk(ADM), checklist(F), chgrp(C), chmod(S), chown(C), configure(ADM) copy(C), cron(C), csh(C), custom(ADM), df(C), diff(C), dircmp(C), disable(C), diskcmp(C), diskcp(C), dmesg(ADM), dos(C), dtype(C), du(C), enable(C), fdisk(ADM), find(C), finger(C), fixperm(ADM), format(C), fsck(ADM), fstab(F), grpcheck(C), init(M), kill(C), login(M), lp(C), lpadmin(ADM), lpinit(ADM), lpstat(C), mail(C), mkdev(ADM), mkuser(ADM), more(C), mount(ADM), netutil(ADM), ps(C), pwadmin(ADM), pwcheck(C), quot(C), rmuser(ADM), shutdown(ADM), sysadmin(ADM), systemid(F), tar(C), ttys(F), umount(ADM), uinstall(ADM), vi(C), wall(ADM), who(C), write(C)

Notes

A knowledge of *vi*(C) is assumed for file edit selections, although the SCO Lyrix[®] editor is used when available.

Acknowledgements

This utility takes its design from the SCO Lyrix Word Processing System.

Name

telinit, mkinittab - Alternative method of turning terminals on and off.

Syntax

telinit state
mkinittab [ttysfile]...

Description

telinit directs the actions of *init*(M). It is an alternative to using *enable*(C) and *disable*(C) to allow and disallow logins on terminals.

telinit generates a new */etc/ttys* file from the */etc/inittab* file. Only those lines from *inittab*(F) which apply in *state* are converted to their *ttys*(F) equivalent. *init* is then signaled to allow or disallow logins on terminals according to */etc/ttys*.

The recognized *state* arguments are:

0-6

Generate */etc/ttys* using the lines in */etc/inittab* which apply to the specified *state*.

q, Q

Do not generate a new */etc/ttys* file, but signal *init* to examine the existing */etc/ttys* file.

s, S

Signal *init* to enter System Maintenance (single-user) mode.

Only the superuser can run *telinit*. Users currently logged onto terminals that are disabled are abruptly killed. Logins are not allowed on terminals not listed in */etc/ttys*.

mkinittab writes on the standard output an *inittab*-format file generated from the specified *ttysfiles*. Each *ttysfile* must be in *ttys* format. If no *ttysfile* is specified, the standard input is read.

Files

/etc/ttys

/etc/inittab

See Also

disable(C), enable(C), getty(M), init(M), inittab(F), login(M), ttys(F)

Notes

inittab is provided for users more familiar with the *telinit* approach to terminal administration, as opposed to the standard XENIX *enable* and *disable* approach. It is intended that a full integration of these two approaches will be provided in a future version of XENIX.

Name

`umount` - Dismounts a file structure.

Syntax

`/etc/umount special-device`

Description

`umount` announces to the system that the removable file structure previously mounted on device *special-device* is to be removed. Any pending I/O for the file system is completed, and the file structure is flagged clean. For a detailed explanation of the mounting process, see `mount(ADM)`.

Files

`/etc/mnttab` Mount table

See Also

`mount(ADM)`, `mount(S)`, `mnttab(F)`

Diagnostics

device busy An executing process is using a file on the named file system



Name

uuccheck - Checks the uucp directories and permissions file.

Syntax

```
/usr/lib/uucp/uuccheck [ -v ] [ -x debug_level ]
```

Description

uuccheck checks for the presence of the *uucp* system required files and directories. It also checks for some obvious errors in the Permissions file (*/usr/lib/uucp/Permissions*). When executed with the *-v* option, it gives a detailed explanation of how the uucp programs will interpret the Permissions file. The *-x* option is used for debugging. *debug-option* is a single digit in the range 1-9; the higher the value, the greater the detail.

Note that *uuccheck* can only be used by the super-user or *uucp*.

Files

```
/usr/lib/uucp/Systems  
/usr/lib/uucp/Permissions  
/usr/lib/uucp/Devices  
/usr/lib/uucp/Maxuuscheds  
/usr/lib/uucp/Maxuuxqts  
/usr/spool/uucp/*  
/usr/spool/uucppublic/*
```

See Also

uucico(ADM), uusched(ADM), uucp(C), uustat(C), uux(C)

Notes

The program does not check file/directory modes or some errors in the Permissions file such as duplicate login or machine name.



Name

uucico - File transport program for the uucp system.

Syntax

```
/usr/lib/uucp/uucico [ -r role_number ] [ -x debug_level ]  
[ -i interface ] [ -d spool_directory ] [ -s ] [ -S ] system_name
```

Description

uucico is the file transport program for *uucp* work file transfers. Role numbers for the **-r** are the digit 1 for master mode or 0 for slave mode (default). The **-r** option should be specified as the digit 1 for master mode when *uucico* is started by a program or *cron*. *uux* and *uucp* both queue jobs that will be transferred by *uucico*. It is normally started by the scheduler, *uusched*, but can be started manually; this is done for debugging. For example, the shell *uutry* starts *uucico* with debugging turned on. A single digit must be used for the **-x** option with higher numbers for more debugging.

The **-i** option defines the interface used with *uucico*. This interface only affects slave mode. Known interfaces are UNIX (default), TLI (basic Transport Layer Interface), and TLIS (Transport Layer Interface with Streams modules, read/write); only the default, UNIX, is applicable in this release.

The **-d** option can be used to specify the *spool* directory: the default is */usr/spool/uucp*.

If **-s** is specified, a call to the specified site is made even if there is no work for site *sitename* in the spool directory, but call only when times in the **Systems** file permit it. This is useful for polling sites that do not have the hardware to initiate a connection.

The **-S** option can be used to specify the system name, overriding the call schedule given in the *Systems* file. For example, **-S** can be used to call a system which is said to be "Never" called in the *Systems* file.

Files

```
/usr/lib/uucp/Systems  
/usr/lib/uucp/Permissions  
/usr/lib/uucp/Devices  
/usr/lib/uucp/Maxuuxqts  
/usr/lib/uucp/Maxuuscheds  
/usr/spool/uucp/*  
/usr/spool/uucppublic/*
```

See Also

uusched(ADM), uutry(ADM), cron(C), uucp(C), uustat(C), uux(C).

Name

uuclean - uucp spool directory clean-up.

Syntax

```
/usr/lib/uucp/uuclean [ -Ctime ] [ -Dtime ] [ -Wtime ] [ -Xtime ] [
-mstring ] [ -otime ] [ -ssystem ] [ -xdebug_level ]
```

Description

uuclean will scan the spool directories for old files and take appropriate action to remove them in a useful way:

Inform the requestor of send/receive requests for systems that can not be reached.

Return mail, which cannot be delivered, to the sender.

Delete or execute mnews for mnews type files (depending on where the news originated--locally or remotely).

Remove all other files.

In addition, there is provision to warn users of requests that have been waiting for a given number of days (default 1). Note that *uuclean* will process as if all option *times* were specified to the default values unless *time* is specifically set.

The following options are available.

- Ctime* Any **C.** files greater or equal to *time* days old will be removed with appropriate information to the requestor. (default 7 days)
- Dtime* Any **D.** files greater or equal to *time* days old will be removed. An attempt will be made to deliver mail messages and execute mnews when appropriate. (default 7 days)
- Wtime* Any **C.** files equal to *time* days old will cause a mail message to be sent to the requestor warning about the delay in contacting the remote. The message includes the *JOBID*, and in the case of mail, the mail message. The administrator may include a message line telling whom to call to check the problem (**-m** option). (default 1 day)
- Xtime* Any **X.** files greater or equal to *time* days old will be removed. The **D.** files are probably not present (if they were, the **X.** could get executed). But if there are **D.** files,

they will be taken care of by D. processing. (default 2 days)

-mstring This line will be included in the warning message generated by the **-W** option.

-otime Other files whose age is more than *time* days will be deleted. (default 2 days) The default line is "See your local administrator to locate the problem".

-ssystem Execute for *system* spool directory only.

-xdebug_level

The **-x** debug level is a single digit between 0 and 9; higher numbers give more detailed debugging information.

This program is typically started by the shell *uudemon.clean*, which should be started by *cron(C)*. *uuclean* can only be executed by the super user or *uucp*.

Files

/usr/lib/uucp directory with commands used by *uuclean* internally

/usr/spool/uucp spool directory

See Also

cron(C), *uucp(C)*, *uux(C)*.

Name

uinstall - Administers UUCP control files.

Syntax

`/etc/uinstall [-r]`

Description

The *uinstall* program is used to manage the content of the control files used by the *uucp* communications system. It allows the user to change the contents of these files without using a text editor. The user need not know the detailed format of each of the control files, although he must be familiar with the function of the various fields within the files. These details are explained in the *XENIX System Administrator's Guide* .

The *uinstall* program can only be executed by the super-user. When invoked with the optional **-r** flag, *uinstall* will not allow any of the files to be modified whether or not the user has made changes to the files.

If *uinstall* finds any of the required **uucp** control files missing from the system, it will create them with the correct access permissions and ownership.

Files

`/etc/systemid`
`/usr/lib/uucp/Systems`
`/usr/lib/uucp/Permissions`
`/usr/lib/uucp/Devices`

See Also

`mkuser(ADM)`



Name

uusched - The scheduler for the uucp file transport program.

Syntax

```
/usr/lib/uucp/uusched [ -x debug_level ] [ -u debug_level ]
```

Description

uusched is the *uucp* file transport scheduler. It is usually started by the daemon *uudemon.hour* that is started by *cron*(C) from an entry in */usr/spool/cron/crontabs/root*:

```
39,9 * * * * /bin/su uucp -c "/usr/lib/uucp/uudemon.hour" > /dev/null
```

The two options are for debugging purposes only; **-x** *debug_level* will output debugging messages from *uusched* and **-u** *debug_level* will be passed as **-x** *debug_level* to *uucico*. The *debug_level* is a number between 0 and 9; higher numbers give more detailed information.

Files

```
/usr/lib/uucp/Systems  
/usr/lib/uucp/Permissions  
/usr/lib/uucp/Devices  
/usr/lib/uucp/Maxuuscheds  
/usr/spool/uucp/*  
/usr/spool/uucppublic/*
```

See Also

uucico(ADM), cron(C), uucp(C), uustat(C), uux(C).



Name

uutry - Tries to contact remote system with debugging on.

Syntax

```
/usr/lib/uucp/uutry [ -x debug_level ] [ -r ] system_name
```

Description

The **uutry** program is a shell that invokes *uucico* to call a remote site. Debugging is automatically enabled at default level 5; **-x** overrides this value. If **uutry** successfully connects to the remote system, **uutry** stores the debugging output in the file */tmp/system*, where *system* is the name of the remote system. In addition, **uutry** uses *tail -f* to print the last 10 lines of the debugging output to the standard output.

To break out of the shell created by **uutry**, press DELETE or BREAK. This returns control to the terminal while *uucico* continues to run, sending the output to */tmp/system_name*.

The **-r** option overrides the retry time in */usr/spool/uucp/status*.

Files

```
/usr/lib/uucp/Systems  
/usr/lib/uucp/Permissions  
/usr/lib/uucp/Devices  
/usr/lib/uucp/Maxuuscheds  
/usr/lib/uucp/Maxuuxqts  
/usr/spool/uucp/*  
/usr/spool/uucppublic/*  
/tmp/system_name
```

See Also

uucico(ADM), uucp(C), uux(C).



Name

uuxqt - Executes remote command requests.

Syntax

```
/usr/lib/uucp/uuxqt [ -s system ] [ -x debug_level ]
```

Description

uuxqt is the program that executes remote job requests from remote systems generated by the use of the *uux* command. (*Mail* uses *uux* for remote mail requests). *uuxqt* searches the spool directories looking for *X* files. For each *X* file, *uuxqt* checks to see if all the required data files are available and accessible, and file commands are permitted for the requesting system. The *Permissions* file is used to validate file accessibility and command execution permission.

There are two environment variables that are set before the *uuxqt* command is executed:

UU_MACHINE is the machine that sent the job (the previous one).

UU_USER is the user that sent the job.

These can be used in writing commands that remote systems can execute to provide information, auditing, or restrictions.

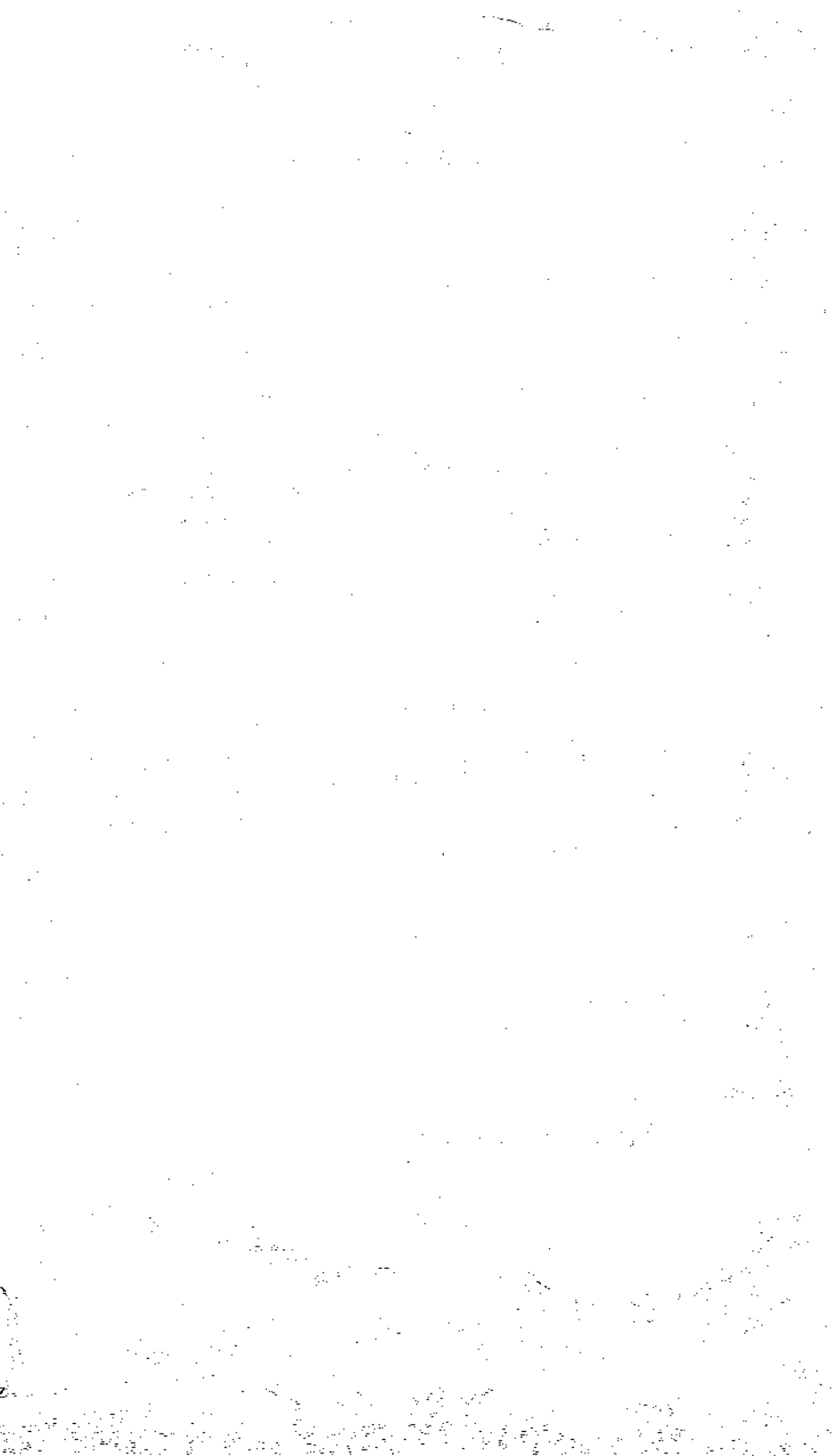
The *-x debug_level* is a single digit between 0 and 9. Higher numbers give more detailed debugging information.

Files

```
/usr/lib/uucp/Permissions  
/usr/lib/uucp/Maxuuxqts  
/usr/spool/uucp/*
```

See Also

uucico(ADM), uucp(C), uustat(C), uux(C), mail(C).



Name

wall - Writes to all users.

Syntax

/etc/wall

Description

wall reads a message from the standard input until an end-of-file. It then sends this message to all users currently logged in preceded by "Broadcast Message from ...". *wall* is used to warn all users, for example, prior to shutting down the system.

The sender should be super-user to override any protections the users may have invoked.

Files

/dev/tty*

See Also

mesg(C), write(C)

Diagnostics

Cannot send to ... The open on a user's tty file has failed.



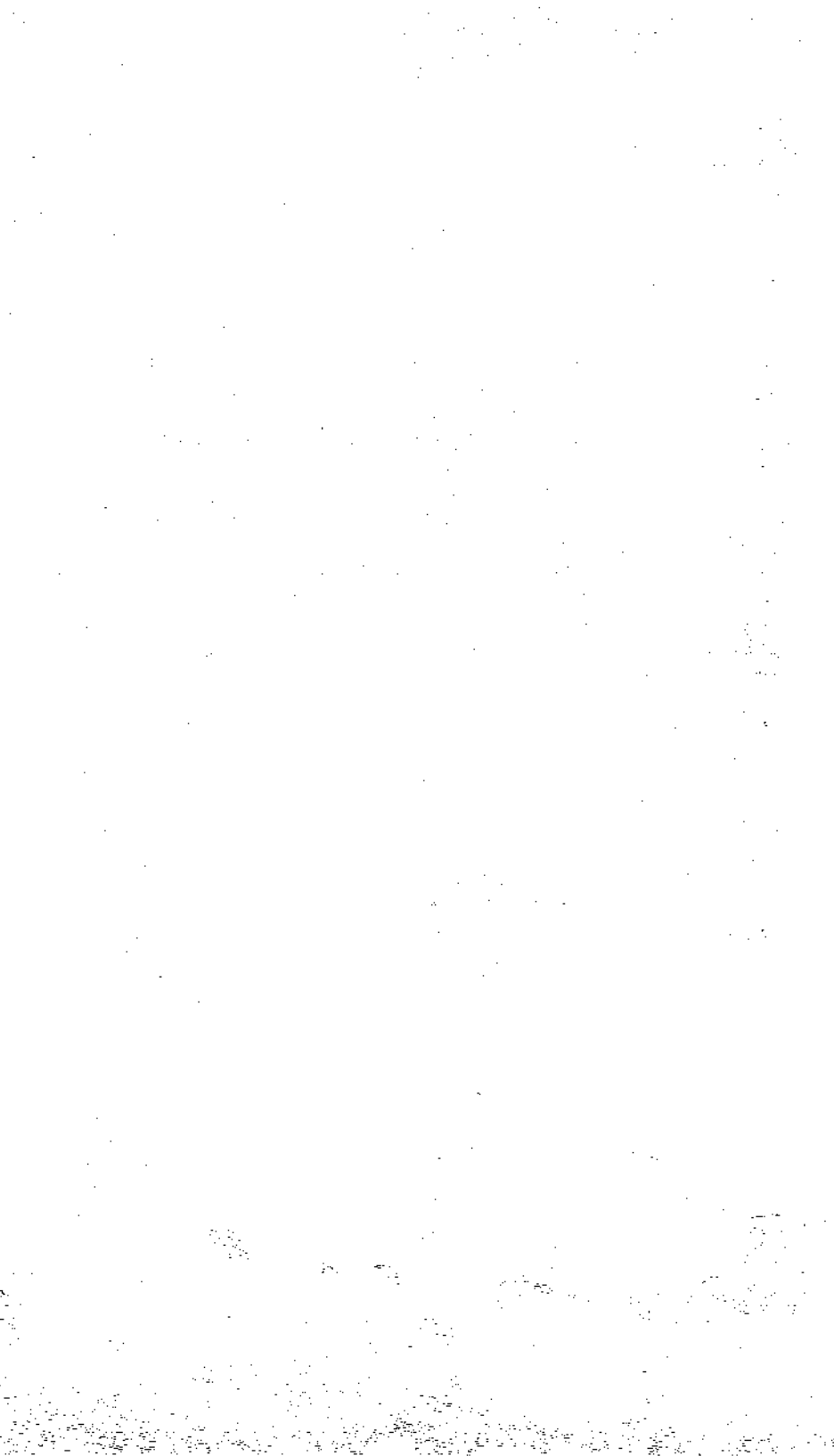
This page intentionally left blank.



Contents

Hardware Dependent (HW)

intro	Introduction to miscellaneous features and files.
8087	Math co-processor for Intel family CPUs.
boot	XENIX boot program.
cmos	Displays and sets the configuration data base.
fd	Floppy devices.
hd	Internal fixed disk drive.
keyboard	Name and function of special keyboard keys.
lp, lp0, lp1, lp2	Line printer device interfaces.,
machine	Description of host machine.
mouse	Mouse or other graphic input device.
parallel	Interface to parallel ports.
ramdisk	Memory block device.
screen, tty[01-n], color, monochrome, ega, pga	Display adapter and video monitor.
scsi	Small computer systems interface.
serial, tty1[a-h], tty1[A-H], tty2[a-h], tty2[A-H]	Interfaces to serial ports.
tape	Cartridge tape device.
terminal	Login terminal.



Name

intro - Introduction to machine related miscellaneous features and files.

Description

The hardware-dependent section (HW) contains information useful in maintaining the system. Included are descriptions of files, devices, tables and programs that are important in maintaining the entire system that are directly related to the kind of computer on which the system runs. This section is intended for use with the 86 family of Intel CPUs, specifically 8086, 8088, 80186, 80286 and 80386 based computers.



Name

8087

Syntax8087
80287**Description**

The 8087 is the INTEL math co-processor for the 8086. The 80287 is the INTEL math co-processor for the 80286. The kernel tests for the presence of an 8087 or 80287 at startup.

If your system has an 8087 or 80287, you must turn off a switch on the main system board in order to enable 8087 interrupts. Check your hardware manual to determine the proper switch and setting. If your system does not have an 8087, or the switch is on, the kernel will run a set of emulator routines which are much slower.

The C compiler available with the program development package generates the appropriate 8087 (or 80287) opcodes. C routines compiled with this compiler have run as much as 200 times as fast as the emulated code. In particular, the standard math library routines run considerably faster if you have an 8087 (or 80287).

The overflow, division by zero, and invalid operand exceptions return a SIGFPE signal. This signal can be caught. The rest of the 8087 and 80287 floating point exceptions (underflow, denormalized operand, and precision error) are masked.

Notes

The emulator returns meaningless information on divide by zero.

There is no obvious way to tell which 8087 (or 80287) exception generated the SIGFPE.



Name

boot - XENIX boot program.

Description

boot is an interactive program used to load and execute standalone XENIX programs. It is used primarily for loading and executing the XENIX kernel, but can load and execute any other programs that are linked for standalone execution. *boot* is a required part of the XENIX Operating System and must be present in the root directory of the root file system to ensure successful loading of the XENIX kernel.

The *boot* program is invoked by the system each time the computer is started. To restart the system without going through lengthy shutdown procedures, you can use the *reboot* command. This causes the system to reboot after shutting down without waiting for keyboard input. See *haltsys (ADM)* for more information.

For diskette boot, the procedure has three stages:

1. The ROMs load the boot block from sector 0 of the floppy, where sector 0 of the disk is the same as sector 0 of the filesystem.
2. The boot block-loads **/boot** from the floppy filesystem.
3. **/boot** executes and prompts the user.

For fixed disk boot, the procedure has five stages:

1. The ROMs load in the *masterboot* block from sector 0 on the hard disk.
2. The *masterboot* block then loads the partition boot block (boot0) from sector 0 of the active partition (see *fdisk(ADM)*).
3. Then, assuming the XENIX partition is active, boot1 is loaded from 1K into the active partition in a 2.2 or later XENIX installation. If the disk was installed with a pre-2.2 XENIX release, then boot1 is assumed to begin at 3K into the active partition. Boot1 spans 20 physically contiguous 1K blocks on the disk.
4. boot1 loads **/boot** from the XENIX file system.
5. **/boot** executes and prompts the user.

/boot and **/xenix** may lie on tracks that have been mapped by *badtrk(ADM)*. *masterboot*, boot0, and boot1 cannot lie on bad tracks.

The fixed disk boot procedure is invoked if the diskette drive is empty.

When first invoked, *boot* prompts for the location of a program to load by displaying the message:

```
XENIX System V
```

```
Boot
```

```
:
```

To specify the location of a program, a device and filename must be given. The filename must include the full pathname of the file containing the standalone program. You can display a list of the current allowable device names by typing a question mark (?).

The format for the device and pathname is as follows:

```
xx(m,o)filename
```

```
or
```

```
xx(m)filename
```

where:

xx = device name

(‘hd’ for the hard disk or ‘fd’ for diskette device)

m = minor device number

(40 for the **root** filesystem on the hard disk)

o = offset in the partition (usually 0). This is optional.

filename = standard XENIX pathname. Must start with a slash if the program is not in the root directory.

All numbers are in decimal. See the manual pages for *hd*(HW) and *fd*(HW) for minor device numbers of these devices. Specifying the offset is optional. The location of the program to be loaded must always be entered first on the command line and be present if other *boot* options are specified either on the command line or in */etc/default/boot*.

If you want *boot* to pause and wait for a <RETURN> before executing the program that it loads, enter the word “prompt” on the command line. For example, if you enter “prompt” and press <RETURN>, *boot* prints the following message and waits for you to press the return key again:

```
Loaded, press <RETURN>.
```

The prompt can be changed to another string as in this example:

```
prompt="change diskettes now"
```

boot loads **xenix** from the diskette, prints the message “change diskettes now”, and waits for <RETURN> to be pressed. No other characters can appear between *prompt*, the “=” sign and the prompt

string, although *string* may contain spaces. When you press <RETURN>, **xenix** will begin execution. "Prompt" can be set either on the command line or in **/etc/default/boot**. If a prompt is not specified, *boot* executes the loaded program without pausing.

If you have just loaded the *boot* program from the distribution diskette, simply press <RETURN> and *boot* defaults to the correct values.

To load XENIX from a hard disk, enter:

```
hd(40,0)xenix
```

To use the default boot string specified in **/etc/default/boot**, simply press <RETURN> when the system displays the boot prompt, and *boot* uses the values specified by DEFBOOTSTR in **/etc/default/boot**.

If nothing is typed after a short while and LOADXENIX is set to YES in the default *root* file system's **/etc/default/boot** file, *boot* times out and behaves as though a <RETURN> had been pressed, except that an "auto" is added to the boot string. (If, in addition to LOADXENIX=YES, TIMEOUT=*n* is defined, *boot* waits *n* seconds before timing out.) *boot* proceeds through the boot procedure, and *init(M)* is passed a **-a** flag with no "prompt".

It is recommended that you install DOS on the hard disk before XENIX. See the manual page for *dos(C)*. However, once you install DOS you can boot it at the XENIX "Boot" prompt by entering "dos".

During XENIX installation, a custom *masterboot* is placed on the hard disk. If a non-standard disk is specified, its parameters are stored and enabled in this *masterboot*.

Configuring The Kernel

boot passes any boot string typed at the boot prompt to the kernel, except for the "prompt" string.

The kernel reads the boot string to determine which peripherals are the root, pipe and swap devices. If no devices are specified in either the **/etc/default/boot** description or on the command line, the default devices compiled into the kernel are used.

Additional arguments in the boot string can alter this default action. These arguments have the form:

```
dev=xx(m,o)
```

or

```
dev=xx(m)
```

where:

dev = The desired system device (**root[dev]**, **pipe[dev]**,
or **swap[dev]**)

xx, m, o = same as for the boot device

If any combination of **root**, **pipe** or **swap** is specified, then those system devices will reside on that device, with the unspecified system devices using the defaults compiled in the kernel. Setting one device does not affect the default values for the other system devices.

Selecting The System Console

You can select the system console at boot time either by entering the command **systty=x** at the boot prompt, or by placing the keywords **SYSTTY=x** in the file **/etc/default/boot**. The letter *x* represents either a number or a string parameter.

If you use the **systty=x** command at boot time, *boot* uses the string parameter *x* to pass the selected console device to the kernel. The values of the boot string parameter **systty** are:

```
sio      Serial port COM1
scrn     Display adapter
```

For example, to assign the system console to the serial port at COM1, enter this command at the boot prompt:

```
systty=sio
```

If you do not specifically set the system console at boot time, the *boot* program follows these steps to determine the system console:

- *boot* reads **/etc/default/boot** and looks for the keywords **SYSTTY=x** where *x* is a number that specifies the system console device.
 - 1** indicates the serial adapter at COM1.
 - 0** indicates the display adapter.
- If **SYSTTY** is not found or **/etc/default/boot** is unreadable, *boot* checks for a display adapter and assigns it as the system console.

- If no display adapter is found, *boot* looks for COM1, sets the serial port to 9600 baud, 8 data bits, 1 stop bit, and no parity, and uses it as the system console.

Thus, to have *boot* automatically set the system console to the serial port at COM1, enter this line in **/etc/default/boot**:

```
SYSTTY=1
```

Aliasing

A set of system devices can be aliased to a single keyword by defining the keyword in the file **/etc/default/boot**. This keyword can then be entered on the "Boot" command line and the boot program then reads the corresponding system devices from **/etc/default/boot** and pass them to the kernel. An alias has the following form:

```
key=file [root=xx(m) pipe=xx(m) swap=xx(m) prompt=["string"]]
```

In all cases, the device specification can also have the format `dev=xx(m,o)`, where `o` is the offset.

For example, if you have a root file system on a second hard disk and want to use it, but want to boot using the **xenix** located on the first hard disk, enter the following line into the **/etc/default/boot** description:

```
disk2=hd(40,0)xenix root=hd(104,0) prompt="Using second disk"
```

The next time you boot the system from the first hard disk, enter "disk2" in response to the "Boot" prompt. **xenix** will be loaded from the first hard disk, and when you see the message, "Using second disk", press <RETURN>. **xenix** will now boot and use the root file system on the second hard disk. Note that you must edit the **/etc/default/boot** file in the root file system on the device from which *boot* will be read, in this case the first hard disk.

Another example: suppose you want to boot off the second drive (hd10) and use the root filesystem and swap space of the second drive. At the boot prompt, use the following bootstring:

```
hd(104)xenix root=hd(104) pipe=hd(104) swap=hd(105)
```

Once booted, you must create the device nodes for the second drive for use by the utilities:

```
fixperm -c -dHD1 /etc/perms/inst
```

Boot options

Boot options can be changed via keywords in **/etc/default/boot**. The following keywords are recognized by *boot*:

LOADXENIX=YES	If YES, <i>boot</i> automatically loads XENIX after a delay time specified by the TIMEOUT parameter. The default value is 60 seconds.
DEFBOOTSTR= <i>string</i>	<i>string</i> is used as the default boot string for timeouts and for no input on the command line. There can be no white space between DEFBOOTSTR, the "=" sign and <i>string</i> .
SYSTTY= <i>x</i>	If <i>x</i> is one (1), the system console device is set to the serial adapter at COM1. If <i>x</i> is zero (0), the system console is set to the main display adapter.
RONLYROOT=NO	Whether or not the root filesystem is to be mounted <i>readonly</i> . This should only be set to "yes" during installation.
FSCCKFIX=YES or NO	Whether or not <i>fsck</i> (ADM) fixes any root system problems by itself. If the variable is set at YES, then <i>fsck</i> (ADM) is run on the root filesystem with the -rr flag.
MULTIUSER=YES or NO	Whether or not <i>init</i> (M) invokes <i>sulogin</i> or proceeds to multiuser mode.
PANICBOOT=YES or NO	Whether or not the system reboots after a <i>panic</i> (<i>)</i> . This variable is read from /etc/default/boot by <i>init</i> .
TIMEOUT= <i>n</i>	<i>n</i> is the number of seconds to wait at boot, before timing out (if LOADXENIX is set to YES).

Diagnostics

If an error occurs, *masterboot* displays an error message, and locks the system. The following is a list of the most common messages and their meanings:

IO ERR

An error occurred when *masterboot* tried to read in the partition boot of the active operating system.

BAD TBL

The bootable partition indicator of at least one of the operating systems in the fdisk table contains an unrecognizable code.

NO OS

There was an unrecoverable error that prevented the active operating system's partition boot from executing.

When *boot* displays error messages, it returns to the "Boot" prompt. The following is a list of the most common messages and their meanings:

bad magic number

The given file is not an executable program.

can't open <pathname>

The supplied pathname does not correspond to an existing file, or the device is unknown.

Stage 1 boot failure

The bootstrap loader cannot find or read the **boot** file. You must restart the computer and supply a file system disk with the **boot** file in the root directory.

not a directory

The specified area on the device does not contain a valid XENIX filesystem.

zero length directory

Although an otherwise valid filesystem was found, it contains a directory of apparently zero length. This most often occurs when a pre- System V XENIX filesystem (with incorrect, or incompatible word ordering) is in the specified area.

fload:read(x)=y

An attempted read of *x* bytes of the file returned only *y* bytes. This is probably due to a premature end-of-file. It could also be caused by a corrupted file, or incorrect word ordering in the header.

Files

/boot
/etc/default/boot
/etc/masterboot
/etc/hdboot0
/etc/hdboot1

See Also

autoboot(ADM), badtrk(ADM), fd(HW), fdisk(ADM), haltsys(ADM), hd(HW), init(M), sulogin(M)

Notes

The computer tries to boot off any diskette in the drive. If the diskette does not contain a valid bootstrap program, errors occur.

The *boot* program cannot be used to load programs that have not been linked for standalone execution. To create standalone programs, the **-A** option of the XENIX linker (*ld(CP)*) and special standalone libraries must be used.

Standalone programs can operate in real or protected mode, but they must not be large or huge models. Programs in real mode can use the input/output routines of the computer's startup ROM.

ONLYROOT should only be set to "yes" for installation. If it is set to "yes" during day-to-day operations, it will prevent you making changes to the root filesystem. You will then be required to boot from the floppy drive, edit the **/etc/default/boot file and reboot.**

Name

cmos - Displays and sets the configuration data base.

Syntax

```
cmos [ address [ value ] ]
```

Description

The *cmos* command displays and/or sets the values in the CMOS configuration data base. This battery-powered data base stores configuration information about the computer that is used at power up to define the system hardware configuration and to direct boot procedures. The data base is 64 bytes long and is reserved for system operation. Refer to your computer hardware manual for more information.

The *cmos* command is typically used to alter the current hardware configuration when new devices are added to the system. When only *address* is given, the command displays the value at that address. If both *address* and a *value* are given, the command assigns the value to that address. If no arguments are given, the command displays the entire contents of the data base.

The CMOS configuration data base may also be examined and modified by reading from and writing to */dev/cmos* file. Because successful system operation depends on correct configuration information, the data base should be modified by experienced system administrators only.

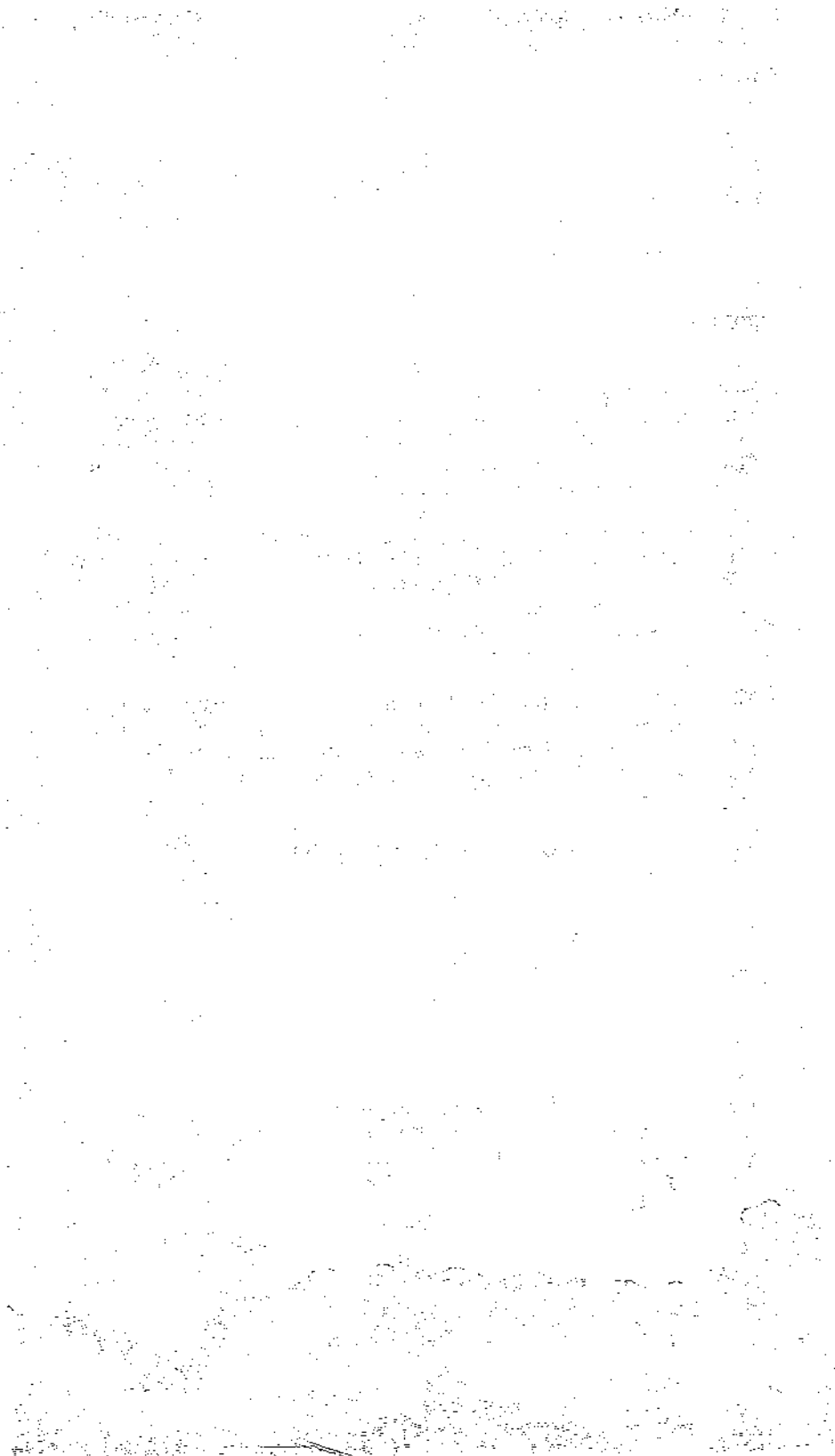
The computer manufacturer's diagnostic diskette should be run before setting the CMOS data base.

Files

```
/etc/cmos  
/dev/cmos
```

Notes

Not all computers have a CMOS configuration data base. Some computers use switches on the main system board to configure the system. Refer to your computer hardware reference manual to determine whether you have a configuration data base.



Name

fd - floppy devices

Description

The **fd** devices implement the XENIX interface with floppy disk drives. Typically, the *tar*(C), *cpio*(C) or *dd*(C) commands are used to read or write floppy disks. For instance,

```
tar tvf /dev/fd0
```

tabulates the contents of the floppy disk in drive 0 (zero).

The block special **fd** devices are also block-buffered. The floppy driver can read or write 1K bytes at a time using raw i/o. Note that block transfers are always a multiple of the 1K disk block size.

The floppy devices are named **/dev/fd0** and **/dev/fd1** (see Notes, below, for more information about device naming procedure).

The corresponding character special (raw) devices, **/dev/rfd0** and **/dev/rfd1**, afford direct, unbuffered transmission between the floppy and the user's read or write transfer address in the user's program.

For information about formatting, see *format*(C).

The minor device number determines what kind of physical device is attached to each device file (see Notes).

Files

/dev/fd0	/dev/rfd048ds8	/dev/rfd096ds15	/dev/rfd0135ds9
/dev/fd1	/dev/rfd148ds8	/dev/rfd196ds15	/dev/rfd1135ds9
/dev/rfd0	/dev/rfd048ds9	/dev/rfd096ds9	/dev/rfd0135ds18
/dev/rfd1	/dev/rfd148ds9	/dev/rfd196ds9	/dev/rfd1135ds18
		/dev/rfd048ss8	
		/dev/rfd148ss9	

Notes

When accessing the character special floppy devices, the user's buffer must begin on a word boundary. The count in a *read*(S), *write*(S), or *lseek*(S) call to a character special floppy device must be a multiple of 1K bytes.

Device names determine the particular drive and media configuration. The device names have the form: fd048ds9 Where: fd0 = drive number (0, 1, 2 or 3) 48 = number of disk tracks per inch (48 or 96) ds = single or double sided floppy (ss or ds) 9 = number of sectors on

the floppy (8 or 9)

For instance, `/dev/fd048ss9` indicates a 48 track per inch, single sided, 9 sector floppy disk device in drive 0.

The minor device numbers for floppy drives depend on the drive and media configuration. The most common are:

Drive	48tpi				96tpi		135tpi	
	ds/8	ds/9	ss/8	ss/9	ds/15	ds/8	ds/9	ds/18
0	12	4	8	0	52	44	36	60
1	13	5	9	1	53	45	37	61
2	14	6	10	2	54	46	38	62
3*								

* reserved for special, non-floppy devices connected to the floppy controller as unit #3.

The scheme for creating minor device numbers is as follows. When interpreted as a binary number, each bit of the minor device number represents some aspect of the device/media configuration.

For example, the minor device number for `/dev/fd048ss8` is "8." Interpreted as a binary number, 8 is:

00001000

This is how each bit, or binary digit, is significant:

48tpi - 0	Sectors per Track		ss - 0	Drive	
96tpi - 1			ds - 1		
135tpi - 1					
32	16	8	4	2	1
0	0	1	0	0	0

Only the last six digits of the number are used in minor device identification. The first significant digit is the third from the left. In this example, the third digit from the left is zero, thus the device is 48tpi. The next two digits mean:

Bits		Sectors per Track
16	8	
0	0	9
0	1	8
1	0	15
1	1	18

The fourth digit tells whether the floppy is single sided (ss - 0) or double sided (ds - 1). The last two signify the drive number:

Bits		Drive Number
2	1	
0	0	0
0	1	1
1	0	2
1	1	3*

* reserved for special, non-floppy devices connected to the floppy controller as unit #3.

Using this information, you can construct any minor device numbers you need.

It is not advisable to format a low density (48tpi) diskette on a high density (96tpi or 135tpi) floppy drive. Low density diskettes written on a high density drive should be read on high density drives. They may or may not be readable on a low density drive.

Use error-free floppy disks for best results on reading and writing.



Name

hd - Internal hard disk drive

Description

Block-buffered access to the *primary* hard disk is provided through the following block special files: **hd00**, **hd01** through **hd04**, **hd0a** and **hd0d**, **root**, and **swap** Block-buffered access to the *secondary* hard disk is provided through the following block special files: **hd10**, **hd11** through **hd14**, **hd1a**,

hd00 refers to the entire physical disk; **hd01** through **hd04** refer to the fdisk partitions. **root** refers to the root file system; **swap** refers to the swap area; The block special files access the disks via the system's normal buffering mechanism and may be read and written without regard to the size of physical disk records.

Character special files follow the same naming convention as the block special files except that the character special file is prefaced with an "r". For example, the character special file referring to the entire physical disk is **/dev/rhd00**.

The following are the names of the fixed disk partitions. Each partition can be accessed through a block interface, for example **/dev/hd01**, or through a character (raw) interface, for example **/dev/rhd01**.

Device File Names for Fixed Disks		
Disk 1	Disk 2	Partition
/dev/hd00 /dev/rhd00	/dev/hd10 /dev/rhd10	entire disk
/dev/hd01 /dev/rhd01	/dev/hd11 /dev/rhd11	first partition
/dev/hd02 /dev/rhd02	/dev/hd12 /dev/rhd12	second partition
/dev/hd03 /dev/rhd03	/dev/hd13 /dev/rhd13	third partition
/dev/hd04 /dev/rhd04	/dev/hd14 /dev/rhd14	fourth partition
/dev/hd0a /dev/rhd0a	/dev/hd1a /dev/rhd1a	active partition
/dev/hd0d /dev/rhd0d	/dev/hd1d /dev/rhd1d	DOS partition
/dev/root /dev/rroot		root file system
/dev/swap /dev/rswap		swap area

Note that the root and swap file names do not exist for a second disk.

To access DOS partitions, specify letters such as "C:" or "D:" to indicate first or second partitions. The file `/etc/default/msdos` contains lines that assign a letter abbreviation for the DOS device name. Refer to *dos(C)*.

The following table lists the minor device number definitions for the hard disk special files, along with examples. Note that the block and character special devices share the same minor device definition. The minor device number definition is as follows: bits 7 and 6 denote physical drive, bits 5-3 denote virtual(*fdisk*) partition and bits 2-0 denote *divvy* partition.

Minor Device Bits				
Phys. 7 6	Virtual 5 4 3	divvy 2 1 0	Device special file name	Description
0 0	0 0 0	0 0 0	/dev/hd00	whole PD 0
0 1	0 0 0	0 0 0	/dev/hd10	whole PD 1
1 0	0 0 0	0 0 0	/dev/hd20	whole PD 2
1 1	0 0 0	0 0 0	/dev/hd30	whole PD 3
0 0	0 0 1	1 1 1	/dev/hd01	PD 0, whole VD 1
0 0	0 1 0	1 1 1	/dev/hd02	PD 0, whole VD 2
0 0	0 1 1	1 1 1	/dev/hd03	PD 0, whole VD 3
0 0	1 0 0	1 1 1	/dev/hd04	PD 0, whole VD 4
0 0	1 0 1	1 1 1	/dev/hd0a	PD 0, whole active VD
0 0	1 1 0	1 1 1	/dev/hd0d	PD 0, whole DOS VD
0 0	1 0 1	0 0 0	/dev/root	PD 0, active virtual, DP 0
0 0	1 0 1	0 0 1	/dev/swap	PD 0, active virtual, DP 1
0 0	1 0 1	0 1 0	/dev/usr	PD 0, active virtual, DP 2
0 0	1 0 1	1 1 0	/dev/recover	PD 0, active virtual, DP 6
0 1	0 0 1	1 1 1	/dev/hd11	PD 1, whole VD 1
0 1	0 1 0	1 1 1	/dev/hd12	PD 1, whole VD 2
0 1	0 1 1	1 1 1	/dev/hd13	PD 1, whole VD 3
0 1	1 0 0	1 1 1	/dev/hd14	PD 1, whole VD 4
0 1	1 0 1	1 1 1	/dev/hd1a	PD 1, whole active VD
0 1	1 1 0	1 1 1	/dev/hd1d	PD 1, whole DOS VD
0 1	1 0 1	0 0 0	/dev/u0	PD 1, active virtual, DP 0†
0 1	1 0 1	0 0 1	/dev/u1	PD 1, active virtual, DP 1†
0 1	1 0 1	0 1 0	/dev/u2	PD 1, active virtual, DP 2†

KEY	VD = virtual drive DP = divvy partition	PD = physical drive † = user-defined name
------------	--	--

The device files **usr** and **u[0-2]** are optional filesystem names; these nodes are not present unless created by the system administrator.

Files

/dev/hd0a	/dev/hd1a	/dev/usr
/dev/rhd0a	/dev/rhd1a	/dev/rusr
/dev/hd0[0-4]	/dev/hd1[0-4]	/dev/root
/dev/rhd0[0-4]	/dev/rhd1[0-4]	/dev/rroot
/dev/hd0d	/dev/hd1d	/dev/swap
/dev/rhd0d	/dev/rhd1d	/dev/rswap

See Also

fdisk(ADM), badtrk(ADM), divvy(ADM), dos(C), mkdev(ADM)

Diagnostics

The following messages are among those that may be printed on the console:

invalid fixed disk parameter table

and:

error on fixed disk (minor *n*), block = *nnnnn*,
cmd=*nnnnn*, status=*nnnn*,
Sector = *nnnnn*, Cylinder/head = *nnnnn*

Possible reasons for the first error include:

- The kernel is unable to get drive specifications, such as number of heads, cylinders, and sectors per track, from the disk controller ROM.
- Improper configuration.
- The disk is not turned on.
- The disk is not supported.

The second error specifies the following information:

- *block* : The XENIX block number within the device.
- *cmd* : The last command sent to the disk controller.
- *status* : The error status from the disk controller.
- *Sector* and *Cylinder/head* specify the location of a possible flaw. This information is used with *badtrk(ADM)*.

Notes

On the first disk, **hd00** denotes the entire disk and is used to access the master boot block which includes the fdisk partition table. For the second disk, **hd10** denotes the entire disk and is used to access its fdisk partition table. Do not write to **hd10** and **hd00**.

Name

keyboard - The PC keyboard.

Description

The PC keyboard is used to enter data, switch screens, and send certain control signals to the computer. XENIX performs terminal emulation on the PC screen and keyboard, and, in doing so, makes use of several particular keys and key combinations. These keys and key combinations have special names that are unique to the XENIX system, and may or may not correspond to the keytop labels on your keyboard. These keys are described later.

When you press a key, one of the following happens:

- An ASCII value is entered
- A string is sent to the computer.
- A function is initiated.
- The meaning of another key, or keys, is changed.

When a key is pressed (a keystroke), the keyboard sends a scancode to the computer, it is interpreted by the keyboard driver. The interpretation of key codes may be modified so that keys can function differently from their default actions.

There are three special occurrences, or keystrokes:

- Switch screens.
- Send signals.
- Change the value of previous character, characters or string.

Switching Screens (Multiscreen)

To get to the next consecutive screen, enter **Ctrl-PrtSc** using the **Ctrl** key, and the **PrtSc** key. Any active screen may be selected by entering **alt-Fn**, where **Fn** is one of the function keys. **F1** refers to the PC display (**/dev/tty01**).

Signals

A signal affects some process or processes. Examples of signals are **Ctrl-d** (end of input, exits from shell), **Ctrl-** (quits a process), **Ctrl-s** (stop output to the screen), and **Ctrl-q** (resume sending output).

Typically, characters are mapped to signals using *stty(C)*. The only way to map signals is using *stty*.

Altering Values

The actual code sent to the keyboard driver can be changed by using certain keys in combination. For example, the SHIFT key changes the ASCII values of the alphanumeric keys. Holding down the **Ctrl** key while pressing another key sends a control code (**Ctrl-d**, **Ctrl-s**, **Ctrl-q**, etc.).

Special Keys

To help you find the special keys, the following table shows which keys on a typical console correspond to XENIX system keys. In this table, a hyphen (-) between keys means 'hold down the first key while pressing the second.'

XENIX Name	Keypop	Action
INTR	Del	Stops current action and returns to the shell. This key is also called the RUB OUT or INTERRUPT key.
BACKSPACE	←	Deletes the first character to the left of the cursor. Note that the "cursor left" key also has a left arrow (←) on its keytop, but you cannot back-space using that key.
Ctrl-d	Ctrl-d	Signals the end of input from the keyboard; also exits current shell.
Ctrl-h	Ctrl-h	Deletes the first character to the left of the cursor. Also called the ERASE key.
Ctrl-q	Ctrl-q	Restarts printing after it has been stopped with Ctrl-s.

Ctrl-s	Ctrl-s	Suspends printing on the screen (does not stop the program).
Ctrl-u	Ctrl-u	Deletes all characters on the current line. Also called the KILL key.
Ctrl-\	Ctrl-\	Quits current command and creates a <i>core</i> file, if allowed. (Recommended for debugging only.)
ESCAPE	Esc	Special code for some programs. For example, changes from insert mode to command mode in the <i>vi</i> (C) text editor.
RETURN	(down-left arrow or ENTER)	Terminates a command line and initiates an action from the shell.
Fn	Fn	Function key <i>n</i> . F1-F12 are unshifted, F13-F24 are shifted F1-F12, F25-F36 are Ctrl-F1 through F12, and F37-F48 are Ctrl-Shift-F1 through F12.

The next *Fn* keys (F49-F60) are on the number pad (unshifted):

F49 - '7'	F55 - '6'
F50 - '8'	F56 - '+'
F51 - '9'	F57 - '1'
F52 - '-'	F58 - '2'
F53 - '4'	F59 - '3'
F54 - '5'	F60 - '0'

For keys F61 through F96, see **`/usr/lib/keyboard/strings`**.

These function keys are not available on all keyboards, but you can map other keys to represent them.

The keyboard mapping is performed through a structure defined in **`/usr/include/sys/keyboard.h`**. Each key can have ten states. The first eight are:

- Base
- Shift
- Ctrl
- Alt
- Ctrl-Shift
- Alt-Shift
- Alt-Ctrl
- Alt-Ctrl-Shift

There are two additional states indicated by two special bytes. The

first is a “special state” byte whose bits indicate whether the key is “special” in one or more of the first eight states.

The second is one of four characters (C, N, B, O) which indicate how the lock keys affect the particular key. This is discussed further in the next section, “Scan Codes.”

Keyboard Mode

Most keyboards normally are in a PC compatibility mode, though some can be put into a native AT keyboard mode. The XENIX utility *kbmode*(ADM) can be used to determine if a keyboard supports AT mode, and can also be used to put the keyboard into AT mode until the next time the system is rebooted. A system can also be configured to boot with the keyboard in AT mode with the *configure*(ADM) utility.

Enhanced keyboards are more fully programmable in AT mode. Also, it recognizes two control keys and an alt key.

Scan Codes

The following table describes the default contents of */usr/lib/keyboard/keys*. The column headings are:

SCAN CODE - The scan code generated by the keyboard hardware when a key is pressed. There is no user access to the scan code generated by releasing a key.

BASE - The normal value of a key press.

SHIFT - The value of a key press when the SHIFT is also being held down.

LOCK - Indicates which lock keys affect that particular key:

- C indicates Capslock
- N indicates Numlock
- B indicates both
- O indicates locking is off

Keys affected by the lock keys C, B, or N, send the shifted value (scan code) of current state when that lock key is on. When the shift key is depressed while a lock key is also on, the key reverts (toggles) to its original state.

The other columns are the values of key presses when combinations of the CTRL, ALT and SHIFT keys are also held down.

All values, except for keywords, are ASCII character values. The keywords refer to the special function keys.

SCAN CODE	BASE	SHIFT	CTRL CTRL	CTRL SHIFT	ALT ALT	ALT SHIFT	ALT CTRL	ALT CTRL SHIFT	LOCK
0	nop	nop	nop	nop	nop	nop	nop	nop	O
1	esc	esc	nop	nop	esc	esc	nop	nop	O
2	'1'	'1'	nop	nop	'1'	'1'	nop	nop	O
3	'2'	'@'	nop	nop	'2'	'@'	nop	nop	O
4	'3'	'#'	nop	nop	'3'	'#'	nop	nop	O
5	'4'	'\$'	nop	nop	'4'	'\$'	nop	nop	O
6	'5'	'%'	nop	nop	'5'	'%'	nop	nop	O
7	'6'	'^'	rs	rs	'6'	'^'	rs	rs	O
8	'7'	'&'	nop	nop	'7'	'&'	nop	nop	O
9	'8'	'*'	nop	nop	'8'	'*'	nop	nop	O
10	'9'	'('	nop	nop	'9'	'('	nop	nop	O
11	'0'	')'	nop	nop	'0'	')'	nop	nop	O
12	'.'	'_'	ns	ns	'.'	'_'	ns	ns	O
13	'='	'+'	nop	nop	'='	'+'	nop	nop	O
14	bs	bs	del	del	bs	bs	del	del	O
15	ht	btab	nop	nop	ht	btab	nop	nop	O
16	'q'	'Q'	dc1	dc1	'q'	'Q'	dc1	dc1	C
17	'w'	'W'	etb	etb	'w'	'W'	etb	etb	C
18	'e'	'E'	enq	enq	'e'	'E'	enq	enq	C
19	'r'	'R'	dc2	dc2	'r'	'R'	dc2	dc2	C
20	't'	'T'	dc4	dc4	't'	'T'	dc4	dc4	C
21	'y'	'Y'	em	em	'y'	'Y'	em	em	C
22	'u'	'U'	nak	nak	'u'	'U'	nak	nak	C
23	'i'	'I'	ht	ht	'i'	'I'	ht	ht	C
24	'o'	'O'	si	si	'o'	'O'	si	si	C
25	'p'	'P'	dle	dle	'p'	'P'	dle	dle	C
26	'['	'{'	esc	esc	'['	'{'	esc	esc	O
27	']'	'}'	gs	gs	']'	'}'	gs	gs	O
28	cr	cr	nl	nl	cr	cr	nl	nl	O
29	ctrl	ctrl	ctrl	ctrl	ctrl	ctrl	ctrl	ctrl	O
30	'a'	'A'	soh	soh	'a'	'A'	soh	soh	C
31	's'	'S'	dc3	dc3	's'	'S'	dc3	dc3	C
32	'd'	'D'	eot	eot	'd'	'D'	eot	eot	C
33	'f'	'F'	ack	ack	'f'	'F'	ack	ack	C
34	'g'	'G'	bel	bel	'g'	'G'	bel	bel	C
35	'h'	'H'	bs	bs	'h'	'H'	bs	bs	C
36	'j'	'J'	nl	nl	'j'	'J'	nl	nl	C
37	'k'	'K'	vt	vt	'k'	'K'	vt	vt	C
38	'l'	'L'	np	np	'l'	'L'	np	np	C
39	','	','	nop	nop	','	','	nop	nop	O
40	'\`	'\`	nop	nop	'\`	'\`	nop	nop	O
41	'\`	'\`	nop	nop	'\`	'\`	nop	nop	O
42	lshift	lshift	lshift	lshift	lshift	lshift	lshift	lshift	O
43	'\`	'\`	fs	fs	'\`	'\`	fs	fs	O
44	'z'	'Z'	sub	sub	'z'	'Z'	sub	sub	C

45	'x'	'X'	can	can	'x'	'X'	can	can	C
46	'c'	'C'	etx	etx	'c'	'C'	etx	etx	C
47	'v'	'V'	syn	syn	'v'	'V'	syn	syn	C
48	'b'	'B'	stx	stx	'b'	'B'	stx	stx	C
49	'n'	'N'	so	so	'n'	'N'	so	so	C
50	'm'	'M'	cr	cr	'm'	'M'	cr	cr	C
51	'<'	'<'	nop	nop	'<'	'<'	nop	nop	O
52	'>'	'>'	nop	nop	'>'	'>'	nop	nop	O
53	'?'	'?'	nop	nop	'?'	'?'	nop	nop	O
54	rshift	rshift	rshift	rshift	rshift	rshift	rshift	rshift	O
55	'*'	'*'	nscr	nscr	'*'	'*'	nscr	nscr	O
56	alt	alt	alt	alt	alt	alt	alt	alt	O
57	' '	' '	' '	' '	' '	' '	' '	' '	O
58	clock	clock	clock	clock	clock	clock	clock	clock	O
59	fkey1	fkey13	fkey25	fkey37	scr1	scr11	scr1	scr11	O
60	fkey2	fkey14	fkey26	fkey38	scr2	scr12	scr2	scr12	O
61	fkey3	fkey15	fkey27	fkey39	scr3	scr13	scr3	scr13	O
62	fkey4	fkey16	fkey28	fkey40	scr4	scr14	scr4	scr14	O
63	fkey5	fkey17	fkey29	fkey41	scr5	scr15	scr5	scr15	O
64	fkey6	fkey18	fkey30	fkey42	scr6	scr16	scr6	scr16	O
65	fkey7	fkey19	fkey31	fkey43	scr7	scr7	scr7	scr7	O
66	fkey8	fkey20	fkey32	fkey44	scr8	scr8	scr8	scr8	O
67	fkey9	fkey21	fkey33	fkey45	scr9	scr9	scr9	scr9	O
68	fkey10	fkey22	fkey34	fkey46	scr10	scr10	scr10	scr10	O
69	nlock	nlock	dc3	dc3	nlock	nlock	dc3	dc3	O
70	slock	slock	del	del	slock	slock	del	del	O
71	fkey49	'7'	'7'	'7'	'7'	'7'	'7'	'7'	N
72	fkey50	'8'	'8'	'8'	'8'	'8'	'8'	'8'	N
73	fkey51	'9'	'9'	'9'	'9'	'9'	'9'	'9'	N
74	fkey52	'.'	'.'	'.'	'.'	'.'	'.'	'.'	N
75	fkey53	'4'	'4'	'4'	'4'	'4'	'4'	'4'	N
76	fkey54	'5'	'5'	'5'	'5'	'5'	'5'	'5'	N
77	fkey55	'6'	'6'	'6'	'6'	'6'	'6'	'6'	N
78	fkey56	'+'	'+'	'+'	'+'	'+'	'+'	'+'	N
79	fkey57	'1'	'1'	'1'	'1'	'1'	'1'	'1'	N
80	fkey58	'2'	'2'	'2'	'2'	'2'	'2'	'2'	N
81	fkey59	'3'	'3'	'3'	'3'	'3'	'3'	'3'	N
82	fkey60	'0'	'0'	'0'	'0'	'0'	'0'	'0'	N
83	del	'.'	del	del	del	del	del	del	N
84	nop	nop	nop	nop	nop	nop	nop	nop	O
85	fkey11	fkey23	fkey35	fkey47	scr11	scr11	scr11	scr11	O
86	fkey12	fkey24	fkey36	fkey48	scr12	scr12	scr12	scr12	O

The following scan codes exist only for keyboards which support, and are in, native AT mode rather than PC compatibility mode.

SCAN CODE	CTRL				ALT		ALT CTRL		LOCK
	BASE	SHIFT	CTRL	SHIFT	ALT	SHIFT	CTRL		
87	fkey11	fkey23	fkey35	fkey47	scr11	scr11	scr11	scr11	O
88	fkey12	fkey24	fkey36	fkey48	scr12	scr12	scr12	scr12	O
89	nop	nop	nop	nop	nop	nop	nop	nop	O
90	nop	nop	nop	nop	nop	nop	nop	nop	O
91	nop	nop	nop	nop	nop	nop	nop	nop	O
92	nop	nop	nop	nop	nop	nop	nop	nop	O
93	nop	nop	nop	nop	nop	nop	nop	nop	O
94	nop	nop	nop	nop	nop	nop	nop	nop	O
95	nop	nop	nop	nop	nop	nop	nop	nop	O
96	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	O
97	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	O
98	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	O
99	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	O
100	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	O
101	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	O
102	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	O
103	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	O
104	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	O
105	del	del	del	del	del	del	del	del	N
106	fkey54	fkey54	fkey54	fkey54	fkey54	fkey54	fkey54	fkey54	O
107	nop	nop	nop	nop	nop	nop	nop	nop	O
108	nop	nop	nop	nop	nop	nop	nop	nop	O
109	nop	nop	nop	nop	nop	nop	nop	nop	O
110	nop	nop	nop	nop	nop	nop	nop	nop	O
111	nop	nop	nop	nop	nop	nop	nop	nop	O
112	nop	nop	nop	nop	nop	nop	nop	nop	O
113	nop	nop	nop	nop	nop	nop	nop	nop	O
114	nop	nop	nop	nop	nop	nop	nop	nop	O
115	nop	nop	nop	nop	nop	nop	nop	nop	O
116	nop	nop	nop	nop	nop	nop	nop	nop	O
117	nop	nop	nop	nop	nop	nop	nop	nop	O
118	nop	nop	nop	nop	nop	nop	nop	nop	O
119	nop	nop	nop	nop	nop	nop	nop	nop	O
120	nop	nop	nop	nop	nop	nop	nop	nop	O
121	nop	nop	nop	nop	nop	nop	nop	nop	O
122	nop	nop	nop	nop	nop	nop	nop	nop	O
123	nop	nop	nop	nop	nop	nop	nop	nop	O
124	nop	nop	nop	nop	nop	nop	nop	nop	O
125	nop	nop	nop	nop	nop	nop	nop	nop	O
126	nop	nop	nop	nop	nop	nop	nop	nop	O
127	nop	nop	nop	nop	nop	nop	nop	nop	O
128	rctrl	rctrl	rctrl	rctrl	rctrl	rctrl	rctrl	rctrl	O

129	ralt	ralt	ralt	ralt	ralt	ralt	ralt	ralt	O
130	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	fkey60	O
131	del	del	del	del	del	del	del	del	N
132	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	fkey49	O
133	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	fkey57	O
134	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	fkey51	O
135	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	fkey59	O
136	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	fkey53	O
137	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	fkey55	O
138	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	fkey50	O
139	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	fkey58	O
140	'/'	nop	nop	nop	'/'	nop	nop	nop	O
141	cr	cr	nl	nl	cr	cr	nl	nl	O

The next table lists the “value” of each of the special keywords used in `/usr/lib/keyboard/keys` (and the preceding table). `mapkey(ADM)` places a “value” in the `ioctl` buffer during key mapping. The keywords are only used in the scan code file (`/usr/lib/keyboard/keys`) for readability.

Name	Value	Meaning
nop	0	No operation - no action from keypress
lshift	2	Left hand shift
rshift	3	Right hand shift
clock	4	Caps lock
nlock	5	Numeric lock
slock	6	Scroll lock
alt	7	Alt key
btab	8	Back tab key - generates fixed sequence (esc [Z)
ctrl	9	Control key
nscr	10	Switch to the next screen
scr1	11	Switch to screen #1
...		...
scr16	26	Switch to screen #16
fkey1	27	Function key #1
...		...
fkey96	122	Function key #96
rctl	128*	Right Control Key
ralt	129*	Right Alt Key

* AT mode keyboard only.

This table lists names and decimal values that are interchangeable in the *mapkey* file. Names are used in place of numeric constants to make it easier to read the scan code table. Again, only the decimal values are placed in the *ioctl* buffer. These are taken from *ascii(M)*.

Name	Value	Name	Value
nul	0	dc1	17
soh	1	dc2	18
stx	2	dc3	19
etx	3	dc4	20
eot	4	nak	21
enq	5	syn	22
ack	6	etb	23
bel	7	can	24
bs	8	em	25
ht	9	sub	26
nl	10	esc	27
vt	11	fs	28
np	12	gs	29
cr	13	rs	30
so	14	ns	31
si	15	del	127
dle	16		

Keyboard Mapping

The PC keyboard is mapped as part of terminal emulation. This kind of mapping is performed only on the computer keyboard, not on remote terminals. Use *mapkey* to change keyboard mapping. To change the mapping for individual channels (multiscreens), use *mapchan(M)*.

Keyboard mapping can also be performed using *ioctl*. The syntax is the same as for string key mapping (see previous section).

For keyboard mapping, *cmd* is `GIO_KEYMAP` to display the current map, and `PIO_KEYMAP` puts the prepared buffer into place.

String Key Mapping

To map string (function) keys, use the *mapstr* (see *mapkey(ADM)*) utility. *mapstr* modifies the string mapping table where function keys are defined.

The string mapping table is an array of 512 bytes (typedef *strmap_t*) containing null terminated strings that redefine the function keys. The first null terminated string is assigned to the first string key, the second string to the second string key, and so on.

There is no limit to the length of any particular string as long as the whole table does not exceed 512 bytes, including nulls. Strings are made null by the introduction of extra null characters.

The following is a list of default function key values:

Default Function Key Values				
Key #	Function	Shift Function	Ctrl Function	Ctrl Shift Function
1	ESC [M	ESC [Y	ESC [k	ESC [w
2	ESC [N	ESC [Z	ESC [l	ESC [x
3	ESC [O	ESC [a	ESC [m	ESC [y
4	ESC [P	ESC [b	ESC [n	ESC [z
5	ESC [Q	ESC [c	ESC [o	ESC [@
6	ESC [R	ESC [d	ESC [p	ESC [[
7	ESC [S	ESC [e	ESC [q	ESC [\
8	ESC [T	ESC [f	ESC [r	ESC []
9	ESC [U	ESC [g	ESC [s	ESC [^
10	ESC [V	ESC [h	ESC [t	ESC [_
11	ESC [W	ESC [i	ESC [u	ESC [`
12	ESC [X	ESC [j	ESC [v	ESC [{

Home	ESC [H	End	ESC [F
Up arrow	ESC [A	Down arrow	ESC [B
Page up	ESC [I	Page down	ESC [G
Left arrow	ESC [D	5	ESC [E
Right arrow	ESC [C	Insert	ESC [L

You can also map string keys using *ioctl*(S). The syntax is:

```
#include <sys/keyboard.h>
ioctl(fd,cmd,buf)
int fd, cmd;
char *buf;
...
```

For string key mapping where *cmd* is *GIO_STRMAP* to display the string mapping table and *PIO_STRMAP* to put the new string mapping table in place.

Files

`/usr/lib/keyboard/keys`
`/usr/lib/keyboard/strings`

See Also

`mapchan(F)`, `mapchan(M)`, `mapkey(ADM)`, `multiscreen(M)`,
`screen(HW)`, `setkey(C)`, `stty(C)`, `kbmode(ADM)`, `configure(ADM)`



Name

lp, lp0, lp1, lp2 - Line printer device interfaces.

Description

The **lp0**, **lp1**, and **lp2** files provide access to the optional parallel ports of the computer. The **lp0** and **lp2** files provide access to parallel ports 1 and 2, respectively. The **lp1** file provides access to the parallel port on the monochrome adaptor.

Only one of **lp0** and **lp1** may be used on a given system. To access two parallel printers on a system, use either **lp0** or **lp1**, and **lp2**.

Files

/dev/lp0
/dev/lp1
/dev/lp2

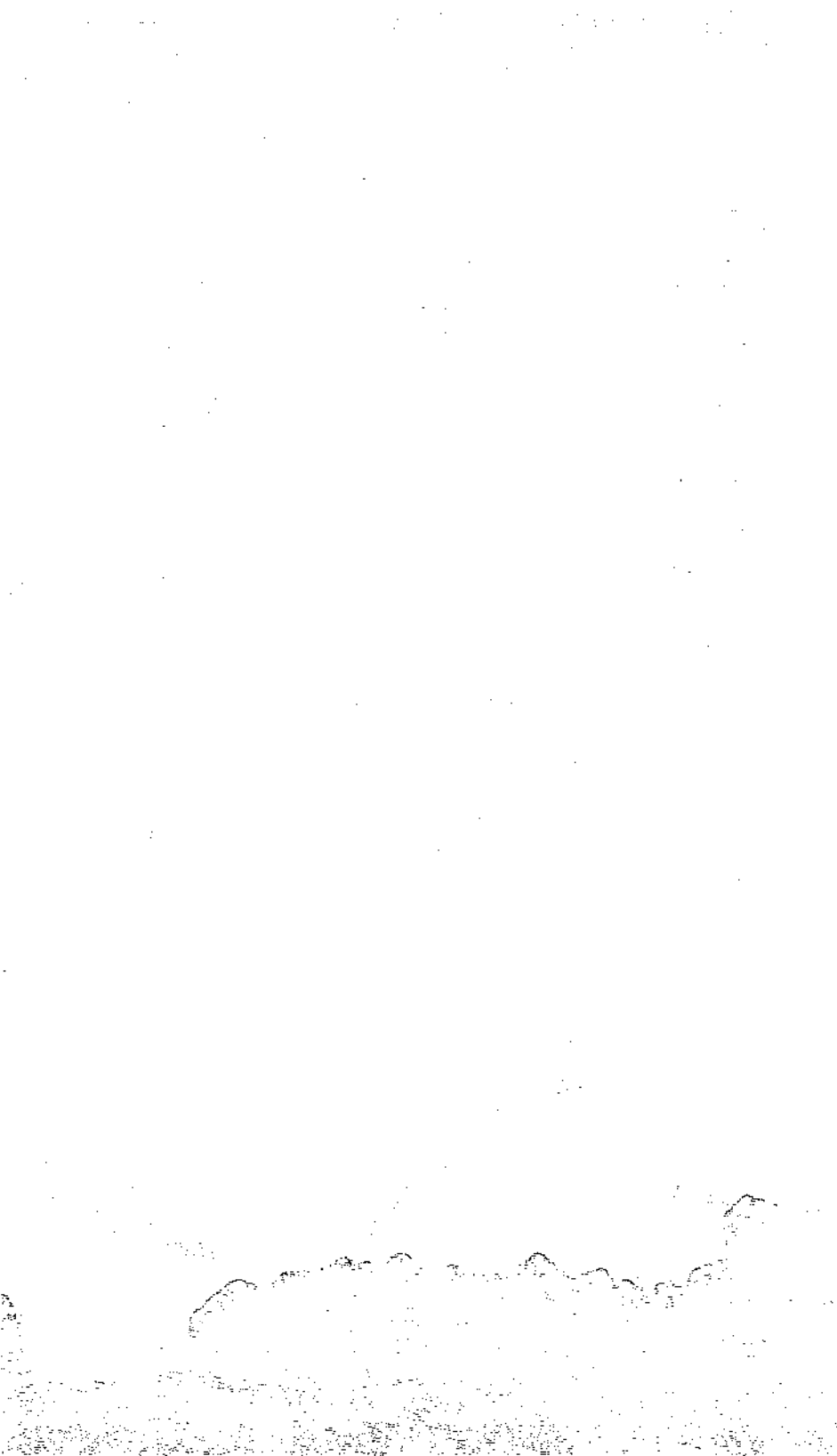
See Also

lp(C), lpadmin(ADM), lpsched(ADM), lpinit(ADM)

Notes

The standard **lp** ports, **lp0**, **lp1**, and **lp2** send a printer initialization string the first time the file is opened after the system is *booted*.

Not all computers have an alternate parallel port slot.



Name

Machine - Description of host machine.

Description

This page lists the internal characteristics of personal computers which use the Intel 8086 processor family and its associated hardware. The information is intended for software developers who wish to transfer relocatable object or executable files from other XENIX machines to a personal computer then prepare the files for execution on the personal computer.

Central Processing Unit	Intel 8086, 8088, 80186, 80286, 80386
Disk Block Size (BSIZE)	1024 bytes
Memory Management Scheme	Unmapped (8086, 8088, 80186) Segmented (80286) Segmented and paged (80386)
Split Instruction and Data	Supported
Variable Stack Size	Supported (8086, 80386 only) (8086, 80386 default configuration)
Fixed Stack Size	Supported (80286 default configuration)
Clock Ticks	.05 second (8086, 8088, 80186) .02 second (80286, 80386)

Binary Compatibility

The small and middle model binary programs created by the C compiler *cc*(CP) run on many processors. The following chart shows which XENIX systems running on which processors produce code executable on other machines. It is assumed that system specific system calls are not used to produce portable code. *cc*(CP) produces code by default, but can also be used as a cross development compiler, by using the appropriate flags.

SCO-*nn* is XENIX distributed by The Santa Cruz Operation, Inc. MS-*nn* is XENIX distributed by Microsoft Corporation. Intel XENIX is distributed by Intel Corporation. Altos XENIX is distributed by Altos Computer Systems. *nn* designates the machine processor. System designates the version of XENIX, either 2,3, 3.0, or System V.

Binary Compatibility			
Your System Processor	Default compiler produces programs which run on System/Processor	Runs default programs created on System/Processor	Compiles (cross development) programs for System/Processor
SCO-86 3.0	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 Sys V	SCO-86 3.0 SCO-186 3.0 Intel, Altos-86 2.3, 3.0	DOS*
SCO-86 System V	SCO-86 Sys V SCO-186 Sys V SCO-286 Sys V MS-286 Sys V	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 Sys V Intel, Altos-86 2.3, 3.0	MS-286 3.0† DOS*
SCO-186 3.0	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 Sys V	SCO-86 3.0 SCO-186 3.0 Intel, Altos-86 2.3, 3.0	DOS*
SCO-186 System V	SCO-86 Sys V SCO-186 Sys V SCO-286 Sys V MS-286 Sys V	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 Sys V Intel, Altos-86 2.3, 3.0	MS-286 3.0† DOS*
SCO-286 3.0	SCO-286 [3.0, Sys V] MS-286 [3.0†, Sys V]	SCO-286 3.0 MS-286 3.0†	DOS*
SCO-286 System V	SCO-86 Sys V SCO-186 Sys V SCO-286 Sys V MS-286 Sys V	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 [3.0, Sys V] MS-286 [3.0†, Sys V]	SCO-286 3.0 MS-286 3.0† DOS*
SCO-386 System V	SCO-86 Sys V SCO-186 Sys V SCO-286 Sys V SCO-386 Sys V MS-286 Sys V MS-386 Sys V	SCO-86 [3.0, Sys V] SCO-186 [3.0, Sys V] SCO-286 [3.0, Sys V] SCO-386 [Sys V] MS-286 [3.0†, Sys V] MS-386 [Sys V]	SCO-286 3.0 MS-286 3.0† DOS*
MS-286 3.0†	MS-286 [3.0†, Sys V] SCO-286 Sys V	SCO-286 3.0	DOS*
MS-286 System V	MS-286 Sys V SCO-286 Sys V	SCO-86 [3.0, Sys V]‡ SCO-186 [3.0, Sys V]‡ SCO-286 [3.0, Sys V]‡	DOS*
MS-386 System V	MS-386 Sys V SCO-386 Sys V	SCO-86 [3.0, Sys V]‡ SCO-186 [3.0, Sys V]‡ SCO-286 [3.0, Sys V]‡ SCO-386 [Sys V]‡	DOS*

* MS-DOS for i8086/8088, i80186 and i80286 processors.

† MS-286 3.0 XENIX is equivalent to Intel 286 3.0 XENIX.

‡ untested, pending release of this product.

See also

clockrate(HW), cc(CP), ld(CP), a.out(F).



Name

mouse - System mouse.

Description

Mouse usage under XENIX is provided through the following special device files:

/dev/mouse	Directory for mouse-related special device files.
/dev/mouse/bus[0-1]	Bus mouse device files.
/dev/mouse/vpix[0-1]	vpix-mouse device files.
/dev/mouse/microsoft_ser	Microsoft serial mouse device files.
/dev/mouse/logitech_ser	Logitech serial mouse device files.
/dev/mouse/mousesys_ser	Mousesys serial mouse device files.
/dev/mouse/ttyp[0-7]	Special pseudo-tty files for mouse input.
/etc/default/usemouse	Default map file for mouse-generated characters.
/usr/lib/event/devices	File containing device information for mice.
/usr/lib/event/ttys	File listing ttys eligible to use mice.
/usr/lib/mouse/*	Alternate map files for mice.

XENIX supports mice attached directly to controller cards on the bus and mice attached to standard serial ports. The command:

mkdev mouse

is used to configure a new mouse or to reconfigure an existing mouse.

See Also

mkdev(ADM), usemouse(C)



Name

parallel - Parallel interface devices.

Description

There are several parallel devices:

/dev/lp0 Main parallel adapter.

/dev/lp1 Adapter on monochrome video card.

/dev/lp2 Alternate parallel adapter (on appropriate machines).

It is not possible to have all three parallel devices on one system. XT computers only allow the use of **/dev/lp0**. Some AT computers allow the use of two parallel devices, **/dev/lp2**, and either **/dev/lp0** or **/dev/lp1**. However, available devices vary from machine to machine, and may instead allow either **/dev/lp0**, or and either **/dev/lp1**, and **/dev/lp2**.

If a parallel device fails to interrupt properly, the parallel driver enters "poll mode." Once interrupts are received from the device, the driver returns to its original mode.

The parallel driver delays a certain amount of time when a parallel device is closed. The amount of delay can affect printer performance, but is necessary to compensate for different sizes of printer buffers and printer speeds. For example, this command sets the delay on close to 1 second, specified in 10ths of a second:

```
stty time 10< /dev/lp0
```

When given from a prompt, this command will only work if the port is open. It is recommended that a variation of this command be placed in the interface script used with the parallel device to achieve the same results:

```
stty time 10 0< &1
```

Notes

Parallel adapters on add-on cards will function, but switches must be set correctly. Some compatible computers have ports lp0 and lp1 reversed.

The *stty*(C) command for output processing is supported on a parallel device. *stty* options that have no effect on a parallel device are ignored and no error messages are displayed.

Usage

Usually invoked by through *lp(C)*, but can be written to directly.

Files

/dev/lp0
/dev/lp1
/dev/lp2

See Also

lp(C), *lp(HW)*, *lpadmin(ADM)*, *lpinit(ADM)*, *lpsched(ADM)*,
serial(HW)

Name

ramdisk - Memory block device

Description

The *ramdisk* device driver provides a block interface to memory. A *ramdisk* can be used like any other block device, including making it into a file systems using *mkfs*(ADM). There are eight *ramdisks* available.

The characteristics of a *ramdisk* file are determined by its minor device number. The bits in the minor device number encode its size, longevity, and which of the eight possible *ramdisks* it is.

The three low-order bits of the minor device number determine which of the eight *ramdisks* is being accessed.

The next four bits of the minor device number determine the size of the *ramdisk*. The size of a *ramdisk* must be a power of 2, and must be at least 16K. Since 4 bits are available, there are 16 possible sizes, starting at 16K and doubling every time the size indicator is incremented, to produce possible sizes of 16K, 32K, 64K, and up.

The high-order bit is a longevity indicator. If set, memory is permanently allocated to that *ramdisk*, and can be deallocated only by rebooting the system. Permanent *ramdisks* can only be allocated by the superuser. However, once a permanent *ramdisk* is allocated (by opening it), it can be read and written by anyone having the appropriate permissions on the *ramdisk* inode.

If clear, the *ramdisk* is deallocated when no processes have it open. To create an easily removable, but semi-permanent *ramdisk*, use a separate process to keep the device open for as long as necessary.

Since a complete set of *ramdisks* (8) would consume 256 inodes, only one example 16K *ramdisk* (*/dev/ram00*) is created when the system is installed. The system administrator can check this existing file to determine the major device number for any other required *ramdisks*. All *ramdisks* will use the same major device number.

The following table shows how the minor device number is constructed:

Example Minor Device Number Construction									
Description	Longevity		Size (see next table)				Ram Disk No.		Minor Device Number
16K (#1) (Temporary)	0	0	0	0	0	0	0	1	1
16K (#1) (Permanent)	1	0	0	0	0	0	0	1	129
64K (#0) (Temporary)	0	0	0	1	0	0	0	0	16
512K (#7) (Permanent)	1	0	1	0	1	1	1	1	175

The contents of the size field and the corresponding *ramdisk* size is shown in the next table.

Size Bits				Ramdisk Size
0	0	0	0	16K
0	0	0	1	32K
0	0	1	0	64K
0	0	1	1	128K
0	1	0	0	256K
0	1	0	1	512K
0	1	1	0	1M
0	1	1	1	2M
1	0	0	0	4M
1	0	0	1	8M
1	0	1	0	16M
1	0	1	1	32M
1	1	0	0	64M
1	1	0	1	128M
1	1	1	0	256M
1	1	1	1	512M

To create a *ramdisk*, follow these steps:

1. Create the device node.

You must first create the device that the ramdisk will reside on. It has the form:

```
mknod device_name b_or_c major_device_number minor_device_number
```

where *b_or_c* "b" or "c". "b" is for blocked devices and is the one you will use. The major number will always be 31. The minor number is derived from the table above. The minor number is the sum of the three attribute columns.

Longevity:

```
permanent = 128 non-permanent = 0
```

Size:

16K = 0	128K = 24	1 Meg = 48	8 Meg = 72
32K = 8	256K = 32	2 Meg = 56	16 Meg = 80
64K = 16	512K = 40	4 Meg = 64	32 Meg = 88

Ram Disk number: 0 through 7 Note: There are only 8 devices available. Two different size devices may not share the same number.

For example, to create a 64K permanent ramdisk, the minor number could vary from 144 to 151. If the disk number was 1 the `mknod` command would be:

```
mknod /dev/ram64 b 31 145
```

2. Make a file system.

This creates a file system on the the ramdisk. In this example `mkfs` has the form:

```
mkfs device_name size_of_file_in_Bsize_blocks
```

In this example, the command to create a 64K file system would be:

```
mkfs /dev/ram64 64
```

3. Mount the filesystem.

This mounts the selected device on the specified mount point. It has the form:

```
mount device_name mount_point
```

In order to mount the example 64K ramdisk on /mnt the command would be:

```
mount /dev/ram64 /mnt
```

To make a file system on a non-permanent *ramdisk*, the device file

must be held open between the *mkfs* and the *mount*(ADM) operations. Otherwise, the *ramdisk* is allocated at the start of the *mkfs* command, and deallocated at its end. Once the *ramdisk* is mounted, it is open until it is unmounted.

The following shell fragment shows one way to use *mkfs* on a non-permanent 512K *ramdisk*, then mount it:

```
( /etc/mkfs /dev/ram40 512
  /etc/mount /dev/ram40 /mnt
) < /dev/ram40
```

Notes

ramdisks must occupy contiguous memory. If free memory is fragmented, opening a *ramdisk* may fail even though there is enough total memory available. Ideally, all *ramdisks* should be allocated at system startup. This helps prevent the *ramdisks* themselves from fragmenting memory.

ramdisks are geared towards use in specialized applications. In many cases, you will notice a *decrease in system performance* when *ramdisks* are used, because XENIX can generally put the memory to better use elsewhere.

Files

/dev/ram00

See Also

mkfs(ADM), mount(ADM), mknod(C)

Name

screen - tty [01-*n*], color, monochrome, ega, vga
- Display adapter and video monitor.

Description

The **tty[01-*n*]** device files provide character I/O between the system and the video display monitor and keyboard. Each file corresponds to a separate teletype device. Although there is a maximum of 12 screens, the exact number available (*n*) depends upon the amount of memory in the computer. The screens are modeled after a 25 line, 80 column ASCII terminal, unless specified otherwise.

System error messages from the kernel are written to **/dev/console**, which is normally the current multiscreen. If the **/dev/console** is the default output device for system error messages, and the display being used is switched to graphics mode, console messages are not displayed. When the video device returns to text mode, a notice message is displayed and the text of the kernel error can be recovered from **usr/adm/messages**.

Although all **tty[01-*n*]** devices may be open concurrently, only one of the corresponding devices can be active at any given time. The active device displays its own screen and takes sole possession of the keyboard. It is an error to attempt to access the **color**, **monochrome**, or **ega** file when no corresponding adapter exists or no multiscreens are associated with it.

To get to the next consecutive screen, enter **Ctrl-PrtSc** using the **Ctrl** key, and the **PrtSc** key. Any active screen may be selected by entering **alt-Fn**, where **Fn** is one of the function keys. For example, **F1** refers to the **tty01** device.

Control Modes

To change the display mode used for the video monitor, open the device file associated with that mode. For example, to switch to display output on an installed CGA, a program should open **/dev/color** and use the selector mapping *ioctl s* on the file descriptor that is returned from the *open* call. Using device files helps ensure future compatibility.

Multiscreens can be reassigned to different adapters (in multi-adapter systems) with these *ioctl*s :

SWAPMONO	Selects the monochrome display as the output device for the video monitor.
SWAPCGA	Selects the regular color display as the output device for the video monitor.
SWAPEGA	Selects the enhanced color display as the output device for the video monitor.
SWAPVGA	Selects the video graphics array color display as the output device for the video monitor.

To find out which display adapter type is currently attached to the video monitor, you can use *ioctl*(S) with the following request:

CONS_CURRENT	Returns the display adapter type currently attached to the video monitor. The return value can be one of: MONO, CGA, EGA, or VGA.
--------------	---

Display Modes

The following *ioctl*s can be used to change the video display mode:

SW_B80x25	Selects 80x25 black and white text display mode. (MONO, CGA, EGA, VGA)
SW_C80x25	Selects 80x25 color text display mode. (CGA, EGA, VGA)
SW_BG320	Selects 320x200 black and white graphics display mode. (CGA, EGA, VGA)
SW_CG320	Selects 320x200 color graphics display mode. (CGA, EGA, VGA)
SW_BG640	Selects 640x200 black and white graphics display mode. (CGA, EGA, VGA)

SW_EGAMONO80x25	Selects EGA (Enhanced Graphics Adapter) mode 7 - emulates support provided by the monochrome display. (EGA, VGA)
SW_EGAMONOAPA	Selects EGA support for 640x350 graphics display mode (EGA mode F). (EGA with mono monitor)
SW_ENHMONOAPA2	Selects EGA mode F*. (EGA with mono monitor)
SW_ENHB40x25	Selects enhanced EGA support for 40x25 black and white text display mode. (EGA, VGA)
SW_ENHC40x25	Selects enhanced EGA support for the 40x25 color text display mode. (EGA, VGA)
SW_ENHB80x25	Selects enhanced EGA support for 80x25 black and white text display mode. (EGA, VGA)
SW_ENHC80x25	Selects enhanced EGA support for 80x25 color text display mode. (EGA, VGA)
SW_CG320_D	Selects EGA support for 320x200 graphics display mode. (EGA mode D.) (EGA, VGA)
SW_CG640_E	Selects EGA support for 640x200 graphics display mode (EGA mode E). (EGA, VGA)
SW_CG640x350	Selects EGA support for 640x350 graphics display mode (EGA mode 10). (EGA, VGA)
SW_ENH_CG640	Selects EGA mode 10*. (EGA, VGA)
SW_MCAMODE	Reinitializes the monochrome adapter. (MONO)
SW_VGA40x25	Selects VGA support for the 40x25 color text display mode (VGA mode 1+). (VGA)
SW_VGA80x25	Selects VGA support for the 80x25 black and white text display mode (VGA mode 2+). (VGA)
SW_VGAM80x25	Selects VGA mode 7+ - emulates support provided by the monochrome display. (VGA with mono monitor)

SW_VGA11	Selects VGA support for the 640x480 graphics display mode (VGA mode 11). (VGA)
SW_VGA12	Selects VGA support for the 640x480 graphics display mode (VGA mode 12). (VGA)
SW_VGA13	Selects VGA support for the 320x200 graphics display mode (VGA mode 13). (VGA)

Switching to an invalid display mode for a display device will result in an error.

Getting Display Modes

The following *ioctl()* requests are provided to obtain information about the current display modes:

CONS_GET	Returns the current display mode setting for current display adapter. (All)
CGA_GET	Returns the current display mode setting of the color graphics adapter. (CGA only)
EGA_GET	Returns the current display mode setting of the enhanced graphics adapter. (EGA only)
MCA_GET	Returns the current display mode setting of the monochrome adapter. (MONO only)
VGA_GET	Returns the current display mode of the video graphics array adapter. (VGA only)

Memory Mapping Modes

The *ioctl(S)* routine is used to map the display memory of the various devices into the user's data space. On the 80286, *ioctl()* returns a selector for the display memory. The macro *sotofar* is used to create a far pointer from this selector so the display memory can be accessed. The *sotofar* macro is located in *usr/include/sys/sysmacros.h*. On the 80386, *ioctl()* returns a (char *).

Refer to your hardware manual for details on various displays, adapters, and controllers.

These *ioctl()* requests can be used to map the display memory:

MAPCONS	Maps the display memory of the adaptor currently being used into the user's data space. (All)
MAPMONO	Maps the monochrome adapter's display memory into the user's data space. (MONO only)
MAPCGA	Maps the color adapter's display memory into the user's data space. (CGA only)
MAPEGA	Maps the enhanced graphics adapter's display memory into the user's data space. (EGA only)
MAPVGA	Maps the video graphics array adapter display memory into the user's data space. (VGA only)

For example, the following code can be used to acquire a pointer to the start of the user data space associated with the color graphics adapter display memory:

```
char far *dp;
int selector;
.
.
.
fd=open ("/dev/color", O_WRONLY);
selector = ioctl (fd, MAPCGA,0);
dp = sotoFar (selector, 0);
.
.
.
```

Note that when the display memory is mapped into the user space, the adapter's start address registers are not set. The start address can be reset in two ways, so that the start address of the display memory corresponds to the upper left hand corner of the screen:

1. Switch modes with an *ioctl()* (the "switch" can be to the present mode). See the "Display Modes" section of this manual page.

2. Change the start address high and low address with the *in-on-port/out-on-port ioctl()*.

The *in-on-port/out-on-port ioctl()*'s can also be used to determine the current value in the start address register, and then set up a pointer to point to the offset in the mapped-in data space.

Graphics Adapter Port I/O

You can use *ioctl(S)* to read or write a byte from or to the graphics adapter port. The *arg* parameter of the *ioctl* call uses the *io_arg* data structure:

```
struct port_io_arg {
    struct port_io_struct args[4];
};
```

As shown above, the *io_arg* structure points to an array of four *port_io* data structures. The *port_io* structure has the following format:

```
struct port_io_struct {
    char    dir; /* direction flag (in vs. out) */
    unsigned_int port; /* port address */
    char    data; /* byte of data */
};
```

You may specify one, two, three, or four of the *port_io_struct* structures in the array for one *ioctl* call. The value of *dir* can be either *IN_ON_PORT* to specify a byte being input to the graphics adapter port or *OUT_ON_PORT* to specify a byte being output to the graphics adapter port. *Port* is an integer specifying the port address of the desired graphics adapter port. *Data* is the byte of data being input or output as specified by the call.

If you are not using any of the *port_io* structures, load the *port* with 0, and leave the unused structures at the end of the array. Refer to the hardware manuals for port addresses and functions for the various adapters.

You can use the following *ioctl(S)* commands to input or output a byte on the graphics adapter port:

MGAIO	Inputs or outputs a byte on the monochrome adapter port as specified. (MONO only)
CGAIO	Inputs or outputs a byte on the color graphics adapter port as specified. (CGA only)

EGAIO	Inputs or outputs a byte on the enhanced graphics adapter port as specified. (EGA only)
VGAIO	Inputs or outputs a byte on the video graphics array adapter port as specified. (VGA only)
CONSIO	Inputs or outputs a byte on the current graphics adapter port as specified. (Any)

To input a byte on any of the graphics adapter ports, load *dir* with `IN_ON_PORT` and load *port* with the port address of the graphics adapter. The byte input from the graphics adapter port will be returned in *data*.

To output a byte, load *dir* with `OUT_ON_PORT`, load *port* with the port address of the graphics adapter, and load *data* with the byte you want output to the graphics adapter port.

Function Keys

`ioctl(S)` can be used to define or obtain the current definition of a function key. The `arg` parameter of the `ioctl` call uses the `fkeyarg` data structure:

```
struct fkeyarg {
    unassigned int keynum;
    char keydef [MAXFK];
    /*Comes from
    char flen; ioctl.h via comcrt.h */
}
```

You can use the following `ioctl(S)` requests to obtain or assign function key definitions:

GETFKEY	Obtains the current definition of a function key. The function key number must be passed in <code>keynum</code> . The string currently assigned to the key will be returned in <code>keydef</code> and the length of the string will be returned in <code>flen</code> when the <code>ioctl</code> is performed.
SETFKEY	Assigns a given string to a function key. The function key number must be passed in <code>keydef</code> and the length of the string (number of characters) must be passed in <code>flen</code> .

Screen Attribute Sequences

The following character sequences are defined by ANSI X3.64-1979 and may be used to control and modify the screen display. Each **Pn** is replaced by the appropriate ASCII number (decimal) to produce the desired effect. The last column is for *termcap* (M) codes, where "n/a" means not applicable.

The use of 7 or 8 bit characters in the escape sequence is a valid invocation for each action defined. For example the ANSI ED command can be invoked via the "ESC [Pn J" (0x1b-0x5b-Pn-0x4a, 7 bit chars) sequence or the "CSI Pn J" (0x9b-Pn-0x4n, 8 bit chars) sequence.

ISO	Sequence	Action	Termcap Code
ED (Erase in Display)	CSI Pn J	Erases all or part of a display. Pn=0 : erases from active position to end of display. Pn=1 : erases from the beginning of display to active position. Pn=2 : erases entire display.	cd
EL (Erase in Line)	CSI Pn K	Erases all or part of a line. Pn=0 : erases from active position to end of line. Pn=1 : erases from beginning of line to active position. Pn=2 : erases entire line.	ce
ECH (Erase Character)	CSI Pn X	Erases Pn characters	n/a
CBT (Cursor Backward Tabulation)	CSI Pn Z	Moves active position back Pn tab stops.	bt

SCREEN (HW)

SCREEN (HW)

SU (Scroll Up)	CSIPn S	Scroll screen up Pn lines, introducing new blank lines at bottom.	sf
SD (Scroll Down)	CSIPn T	Scrolls screen down Pn lines, introducing new blank lines at top.	sr
CUP (Cursor Position)	CSIP1 ; P2 H	Moves active position to location P1 (vertical) and P2 (horizontal).	cm
HVP (Horizontal & Vertical Position)	CSIP1 ; P2 f	Moves active position to location P1 (vertical) and P2 (horizontal).	n/a
CUU (Cursor Up)	CSIPn A	Moves active position up Pn number of lines.	up (ku)
CUD (Cursor Down)	CSIPn B	Moves active position down Pn number of lines.	do (kd)
CUF (Cursor Forward)	CSIPn C	Moves active position Pn spaces to the right.	nd (kr)
CUB (Cursor Backward)	CSIPn D	Moves active position Pn spaces backward.	bs (kl)
HPA (Horizontal Position Absolute)	CSIPn ‘	Moves active position to column given by Pn .	n/a
HPR (Horizontal Position Relative)	CSIPn a	Moves active position Pn characters to the right.	n/a
VPA (Vertical Position Absolute)	CSIPn d	Moves active position to line given by Pn .	n/a

VPR (Vertical Position Relative)	CSI Pn e	Moves active position down Pn number of lines.	n/a
IL (Insert Line)	CSI Pn L	Inserts Pn new, blank lines.	al
ICH (Insert Character)	CSI Pn @	Inserts Pn blank places for Pn characters.	ic
DL (Delete Line)	CSI Pn M	Deletes Pn lines.	dl
DCH (Delete Character)	CSI Pn P	Deletes Pn number of characters.	dc
CPL (Cursor to Previous Line)	CSI Pn F	Moves active position to beginning of line, Pn lines up.	n/a
CNL (Cursor Next Line)	CSI Pn E	Moves active position to beginning of line, Pn lines down.	n/a
SGR (Select Graphic Rendition)	CSI 0 m	Resets bold, blink, blank, underscore, and reverse. Color: Restores normal selected colors.	n/a
SGR	CSI 1 m	Sets bold. Color: Sets intensity (changes <i>color</i> to <i>lt_color</i>).	n/a
SGR	CSI 4 m	Sets underscore. Color: No effect.	n/a
SGR	CSI 5 m	Sets blink. Color: Changes background <i>lt_color</i> to <i>color</i> ; fore- ground blinks.	n/a

SGR	CSI 7 m	Sets reverse video. Color: Uses reverse selected colors.	so
SGR	CSI 10 m	Select primary font.	GE
SGR	CSI 11 m	Select first alternate font. Allows ASCII characters less than 32 to be displayed as ROM characters.	n/a
SGR	CSI 12 m	Select second alternate font. Toggles high bit of extended ASCII code before displaying as ROM characters.	GS
SGR	ESC [4 m	Underscores. Color: No effect	n/a
SGR	CSI 3 C m	Color: Selects foreground color <i>C</i> (see Table 1 below).	n/a
SGR	CSI 4 C m	Color: Selects background color <i>C</i> (see Table 1).	n/a
SGR	CSI 8 m	Sets blank (non-display).	n/a

The following color codes and sequences are defined by International Organization for Standardization ISO DP 6429.

C	Color
0	Black
1	Red
2	Green
3	Yellow
4	Blue
5	Magenta
6	Cyan
7	White

ISO	Sequence	Action	Termcap Code
SM (Set Mode)	ESC [2 h	Lock keyboard. Ignores keyboard input until unlocked. Characters are not saved.	n/a
MC (Media Copy)	ESC [2 i	Send screen to host. Current screen contents are sent to the application.	n/a
RM (Reset Mode)	ESC [2 l	Unlock keyboard. Re-enable keyboard input.	n/a

The following color codes and sequences are additional control sequences.

Cn	Color	Cn	Color
0	Black	8	Grey
1	Blue	9	Lt. Blue
2	Green	10	Lt. Green
3	Cyan	11	Lt. Cyan
4	Red	12	Lt. Red
5	Magenta	13	Lt. Magenta
6	Brown	14	Yellow
7	White	15	Lt. White

Name	Sequence	Action	Termcap Code
n/a	CSI = c A	Set overscan color to color <i>c</i> . <i>c</i> is a decimal value taken from Table 2 above. (This sequence may not be supported on all hardware.)	n/a
n/a	CSI = p ; d B	Set the bell parameter to the decimal values of <i>p</i> and <i>d</i> . <i>p</i> is the period of the bell tone in units of 840.3 nanoseconds, and <i>d</i> is the duration of the tone in units of 100 milliseconds.	n/a

n/a	CSI = <i>s</i> ; <i>e</i> C	Set the cursor to start on scanline <i>s</i> and end on scanline <i>e</i> .	n/a
n/a	CSI = <i>x</i> D	Turn on or off (<i>x</i> =1 or 0) the intensity of the background color.	n/a
n/a	CSI = <i>x</i> E	Set or clear (<i>x</i> =1 or 0) the Blink vs. Bold background bit in the 6845 crt controller.	n/a
n/a	CSI = <i>c</i> F	Set normal foreground color to <i>c</i> . (<i>c</i> is a decimal parameter taken from Table 2.)	n/a
n/a	CSI = <i>c</i> G	Set normal background. (See Table 2.)	n/a
n/a	CSI = <i>c</i> H	Set reverse foreground. (See Table 2.)	n/a
n/a	CSI = <i>c</i> I	Set reverse background. (See Table 2.)	n/a
n/a	CSI = <i>c</i> J	Set graphic foreground. (See Table 2.)	n/a
n/a	CSI = <i>c</i> K	Set graphic background. (See Table 2.)	n/a
n/a	ESC [Pn g	Accesses alternate graphics set. Not the same as "graphics mode." Refer to your owner's manual for decimal/character codes (Pn) and possible output characters.	n/a
n/a	ESC Q Fn 'string'	Define function key Fn with <i>string</i> . String delimiters ' and ' may be any character not in <i>string</i> . Function keys are numbered 0 through 9 (F1 = 0, F2 = 1, etc.).	n/a
n/a	ESC [Pn z CSI Pn z	Pn should be equal to the number of the screen to switch to. Will only work if screen was configured for at boot, else no action will take place.	n/a

Files

/dev/console

/dev/tty [02 -n]

/dev/color

/dev/monochrome

/dev/ega

/dev/vga

See Also

console(M), ioctl(S), keyboard(HW), keymap(M), mapkey(ADM), mapchan(M), multiscreen(M), setcolor(C), stty(C), systty(M), vidi(C), termcap(M), tty(M)

Name

scsi - small computer systems interface.

Description

SCSI provides a standard interface for peripherals such as hard disks, printers, tape drives and others. SCSI is run via a host adapter card that can support up to 8 controllers, each supporting up to 8 devices. Note that only 4 hard disks can be attached to a controller.

The minor device numbering scheme for SCSI devices (for example, a hard disk) is the same as the standard minor device number scheme for non-SCSI devices.

Each SCSI controller has its own major device number.

Notes

This utility is not applicable to all hardware/software configurations and may not be included in your distribution.
--

Currently, the only SCSI host adapter card supported is the Adaptec AHA-1540 controlling 1 or 2 hard disks and 1 tape drive.

See Also

hd(HW), tape(HW)



Name

tty1[a-h] , tty1[A-H] , tty2[a-h] , tty2[A-H] - Interface to serial ports

Description

The **tty1[a-h]**, **tty1[A-H]**, **tty2[a-h]** and **tty2[A-H]** files provide access to the standard and optional serial ports of the computer. Each file corresponds to one of the serial ports (with or without modem control). Files are named according to the following conventions:

- The first number in the file name corresponds to the COM expansion slot.
- Lower case letters indicate no modem control.
- Upper case letters indicate the line has modem control.

tty1a and **tty1A** both refer to COM 1, whereas **tty2a** and **tty2A** both refer to COM 2.

For example, with a four port expansion board installed at COM 1 and a single port board installed at COM 2, you can access:

tty1a	tty1A
tty1b	tty1B
tty1c	tty1C
tty1d	tty1D
tty2a	tty2A

Each serial port has modem and non-modem invocations. The device names in the following table refer to the serial ports, with and without modem control. The first section of the table describes boards at COM 1 and the second section describes boards installed at COM 2. "Minor" is the minor device number for the port (see *mknod(C)*).

Serial Lines						
Board Type	Non-Modem Control		Modem Control			
	Minor	Name	Minor	Name		
1 Port	0	tty1a	128	tty1A		
	4 Port	1	tty1b	129	tty1B	
		2	tty1c	130	tty1C	
		3	tty1d	131	tty1D	
	8 Port	4	tty1e	132	tty1E	
		5	tty1f	133	tty1F	
		6	tty1g	134	tty1G	
		7	tty1h	135	tty1H	
1 Port	8	tty2a	136	tty2A		
	4 Port	9	tty2b	137	tty2B	
		10	tty2c	138	tty2C	
		11	tty2d	139	tty2D	
	8 Port	12	tty2e	140	tty2E	
		13	tty2f	141	tty2F	
		14	tty2g	142	tty2G	
		15	tty2h	143	tty2H	

Interrupt Vectors:

All board(s) installed at COM 1 - 4
 All board(s) installed at COM 2 - 3

For a list of I/O addresses, see the *Release Notes* furnished with your distribution.

Access

The files may only be accessed if the corresponding serial interface card is installed and its jumper I/O address correctly set. Also, for multi-port expansion cards, you must use the *mkdev(ADM)* program to create more than the default number of files.

The serial ports must also be defined in the system configuration. Check your hardware manual to determine how your system is configured, via a CMOS database or by switch settings on the main system board. If your system is configured using a CMOS database, the ports are defined in the database (see *cmos(HW)*). Otherwise, define the ports by setting the proper switches on the main system board. Refer to your computer hardware manual for switch settings.

It is an error to attempt to access a serial port that has not been installed and defined.

The serial ports can be used for a variety of serial communication purposes such as connecting login terminals to the computer, attaching printers, or forming a serial network with other computers. Note that a serial port may operate at most of the standard XENIX baud rates, and that the ports (on most computers) have a DTE (Data Terminal Equipment) configuration. The following table defines how each pin is used for 25-pin and 9-pin connections:

25-Pin	9-Pin	Description
2	2	Transmit Data
3	3	Receive Data
4	7	Request to Send
5	8	Clear to Send
7	5	Signal Ground
8	1	Carrier Detect (Data Set Ready)
20	4	Data Terminal Ready

Only pins 2, 3, and 7 (2,3 and 5 for 9-pin) are necessary for a terminal (or direct) connection.

A modem control device (port) uses pins 2, 3, and 7 in the same way as a non-modem control device: send on pin 2 and receive on pin 3. Pin 7 is data ground. On a non-modem control device the state of the other pins are not set or read. On a modem control device, pins 4 and 20 (RTS & DTR) are asserted and the port will not open until pin 8 (CXD) is asserted. That is, no signal travels from pin 2 until pin 8 is asserted from another source. The modem control device monitors the the status of pin 8.

See *tty*(M) and *termio*(M) for the details of serial line operation in the XENIX system.

Files

```
/dev/tty1[a-h]
/dev/tty1[A-H]
/dev/tty2[a-h]
/dev/tty2[A-H]
```

See Also

cmos(HW), *csh*(C), *cu*(C), *getty*(ADM), *mkdev*(ADM), *mknod*(C), *nohup*(C), *open*(S), *termio*(M), *tty*(M), *uucp*(C)

Notes

If you login via a modem control serial line, hanging up logs that line out and kills your background processes. See *nohup(C)* and *csH(C)*.

You cannot use the same serial port with both modem and non-modem control at the same time. For example, you cannot use *ty1a* and *ty1A* simultaneously.

Use a modem cable to connect your modem to a computer.

Name

tape - Cartridge tape device.

Description

The *tape* device implements the XENIX interface with a tape drive. Typically, the *tar*(C), *cpio*(C), *dd*(C), *backup*(C), *dump*(C), or *restore*(C) commands are used to access a tape drive.

A single tape drive with a raw (character, non-blocking) interface is supported. There are three standard tape device types. Devices beginning with the "r" prefix, (for "raw device"), should be used for most normal tape work, while devices with the "n" prefix, ("for no rewind on hold"), should be used for storing and restoring multiple files. Devices beginning with the "e" prefix (for ECC device) support a 2/64 error recovery scheme. Thus two 512-byte blocks out of every 64 blocks can be bad and the driver will correct the errors. This software ECC support provides a high degree of error recovery.

The following table summarizes the base naming conventions for the tape drives supported:

ct0,1	QIC24 unit 0,1
ct2,3	QIC11 unit 0,1
ctmini	floppy controller-based cartridge tape
mt0,1	reel to reel unit 0,1 1600 bpi
mt2,3	reel to reel unit 0,1 800 bpi
mt4,5	reel to reel unit 0,1 6250 bpi

tape(C) describes the commands used to access tape drives.

Files

```
/dev/rct0
/dev/nrct0
/dev/rct2
/dev/nrct2
/dev/erct0
/dev/rctmini
```

Notes

After certain tape operations are executed, the system returns a prompt before the tape controller has finished its operation. If the user enters another tape command too quickly, a "device busy" error is returned until the tape device is finished with its previous operation.

Periodic tape cartridge retensioning and tape head cleaning are necessary for continued error-free operation of the tape subsystem. Use *tape*(C), to retension the tape.

See Also

backup(C), *cpio*(C), *dd*(C), *dump*(C), *format*(C), *tape*(C), *tar*(C), *restore*(C).

Name

terminal - Login terminal.

Description

A *terminal* is any device used to enter and display data. It may be connected to the computer:

- By a serial wire, either direct or dialup
- As a virtual terminal, for example with emulator software
- Through a display adapter

A terminal has an associated device file `/dev/tty*`.

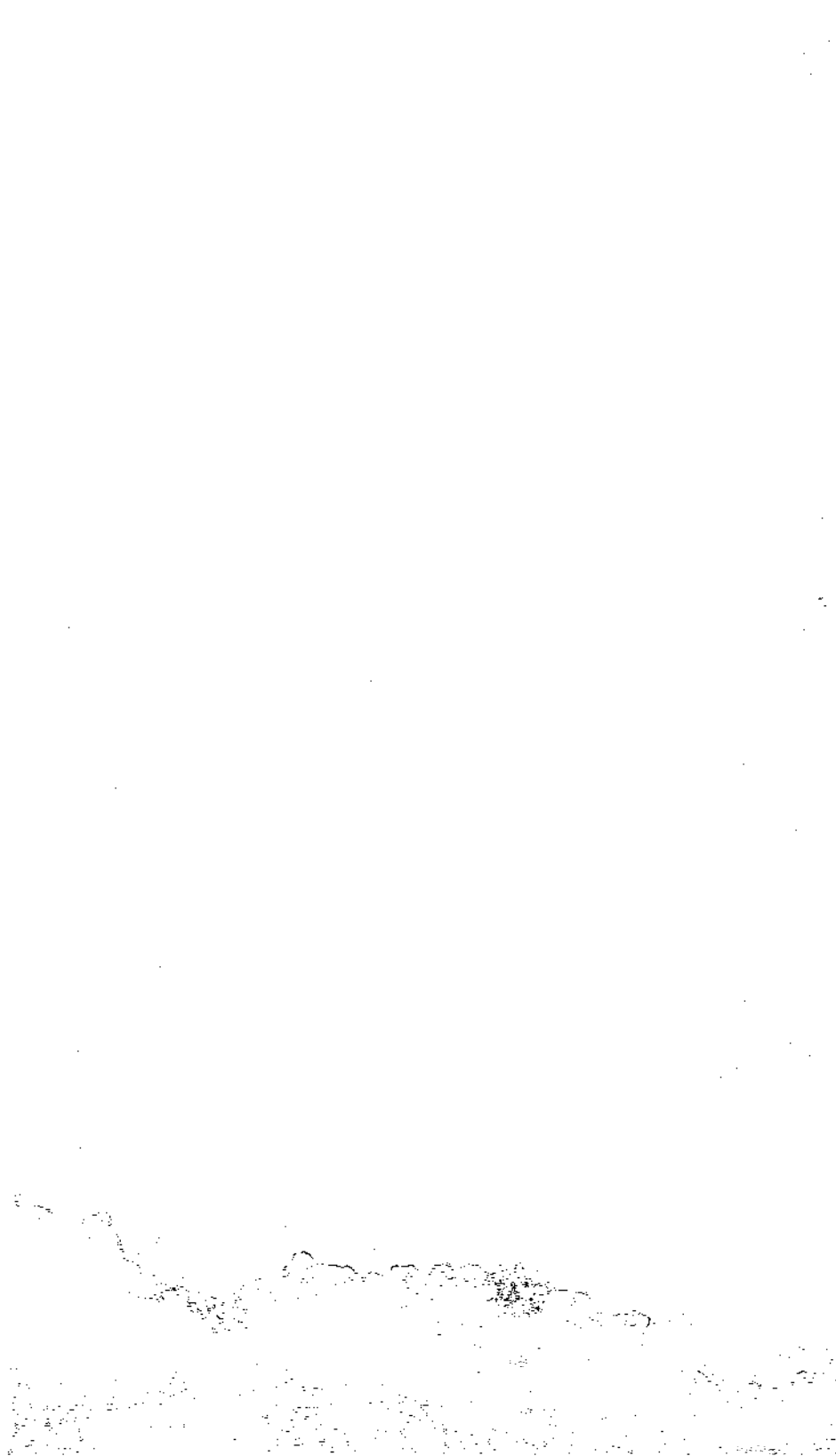
Files

`/dev/tty*`

See Also

`console(M)`, `disable(C)`, `enable(C)`, `mkdev(ADM)`, `serial(HW)`, `stty(C)`, `vidi(C)`, `termcap(M)`, `term(F)`, `terminals(M)`













AU01200P000
26103